Evolving Hyperboxes for Enhanced Classification and Scalable Feature Subset Selection

A thesis submitted during 2022 to the University of Hyderabad in partial fulfillment of the award of a Ph.D. degree in School of Computer and Information Sciences

by

Anil Kumar

Reg. No: 15MCPC10

School of Computer and Information Sciences
University of Hyderabad

(P.O.) Central University, Gachibowli,
Hyderabad – 500046
Telangana, India

2022

CERTIFICATE

This is to certify that the thesis entitled **"Evolving Hyperboxes for Enhanced Classification and Scalable Feature Subset Selection"** submitted by **Anil Kumar** bearing **Reg. No: 15MCPC10** in partial fulfillment of the requirements for the award of **Doctor of Philosophy** in **Computer Science** is a bonafide work carried out by him under my supervision and guidance.

The thesis is free from plagiarism and has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma.

The student has the following publications before submission of the thesis for adjudication and has produced evidence for the same in the form of acceptance letter or the reprint in the relevant area of his research:

1. **Anil Kumar** and P.S.V.S. Sai Prasad. **Hybridization of Fuzzy Min-Max Neural Networks with kNN for Enhanced Pattern Classification**. Singh M., Gupta P., Tyagi V., Flusser J., Ören T., Kashyap R. (eds): *Proceedings of Advances in Computing and Data Sciences. ICACDS 2019.*, Vol. 1045, Pages 32-44, CCIS 168, Springer 2019, ISBN 978-981-13-9938-1. https://doi.org/10.1007/978-981-13-9939-8_4 **(Indexed in SCOPUS)**. The work reported in this publication appears in Chapter 3.

2. **Anil Kumar** and P. S. V. S. Sai Prasad. **Scalable Fuzzy Rough Set Reduct Computation Using Fuzzy Min–Max Neural Network Preprocessing**. Vol. 28, Pages 953-964, May 2020 *IEEE Transactions on Fuzzy Systems*. http://dx.doi.org/10.1109/TFUZZ.2020.2965899 **(Indexed in SCI, SCOPUS)**. The work reported in this publication appears in Chapter 4.

3. **Anil Kumar** and P. S. V. S. Sai Prasad. **Enhancing the Scalability of Fuzzy Rough Set Approximate Reduct Computation Through Fuzzy Min-Max Neural Network and Crisp Discernibility Relation Formulation**. Vol. 110,

Pages 104697, Apr 2022 *Engineering Applications of Artificial Intelligence*. https://doi.org/10.1016/j.engappai.2022.104697 **(Indexed in SCI, SCOPUS)**. The work reported in this publication appears in Chapter 5.

4. **Anil Kumar** AND P. S. V. S. SAI PRASAD. **Incremental Fuzzy Rough Sets based Feature Subset Selection using Fuzzy Min-Max Neural Network Preprocessing**. Vol. 139, Pages 69-87, Dec 2021 *International Journal of Approximate Reasoning*. https://doi.org/10.1016/j.ijar.2021.09.006 **(Indexed in SCI, SCOPUS)**. The work reported in this publication appears in Chapter 6.

Further the student has passed the following courses towards fulfilment of course work requirement for Ph.D.

Course Code	**Name of the Course**	**Credits**	**Pass/Fail**
CS-801	Data Structures and Algorithms	4	Pass
CS-802	Operating Systems and Programming	4	Pass
AI-821	Trends in Soft Computing	4	Pass
AI-851	Pattern Recognition	4	Pass

Dr. P.S.V.S. Sai Prasad
(Supervisor)
School of Computer and
Information Sciences,
University of Hyderabad
Hyderabad - 500046, India

(Dean)
School of Computer and
Information Sciences,
University of Hyderabad
Hyderabad - 500046, India

DECLARATION

I, **Anil Kumar**, hereby declare that this thesis entitled **"Evolving Hyperboxes for Enhanced Classification and Scalable Feature Subset Selection"** submitted by me under the guidance and supervision of **Dr. P. S. V. S. Sai Prasad** is a bonafide research work and is free from any plagiarism. I also declare that it has not been submitted previously in part or in full to this University or any other University or Institution for the award of any degree or diploma. I hereby agree that my thesis can be submitted in Shodganga/INFLIBNET.

A report on plagiarism statistics from the University Librarian is enclosed.

Date :

Signature of the Student

Name: Anil Kumar

Reg. No.: 15MCPC10

*To my parents, **Narayan Chand Prasad and Sona Devi**, and Siblings **Arun Kumar and Dipshikha Kumari** without whose support and encouragement, this would not have been possible.*

*To my supervisor, **Dr. P.S.V.S. Sai Prasad**, without whose support and encouragement, this would not have been possible.*

Acknowledgements

I take this opportunity to thank all those who have helped me in the completion of my work leading to this thesis and in making my stay at University of Hyderabad, memorable.

First, I am ever indebted to this university for giving me an opportunity to undertake this exciting and challenging journey. With an ideal mix of science and arts, aiming for excellence and developing a holistic individual, this university has helped me grow as a human being with its beautiful environment, perfect ambience and remarkable people. I will always cherish the time I spent here at the university.

I would like to express my sincere gratitude and indebtedness to my supervisor, **Dr. P.S.V.S Sai Prasad**, for his constant guidance and motivation throughout my PhD journey. His deep and comprehensive knowledge of the subjects and persistent efforts helped me to shape and substance to this research work.

I would like to acknowledge RAC members **Prof. Salman Abdul Moiz** and **Prof. V. Ch. Venkaiah** for their valuable suggestions for improving the thesis.

I owe special thanks to **Prof. Arun K. Pujari**, who motivated me to do further study as PhD after my master's in artificial intelligence.

I would like to express my sincere thanks to **Prof. Chakravarthy Bhagvati**, Dean, School of Computer and Information Sciences (SCIS), for his cooperation.

I would like to thank **Prof. Atul Negi**, School of Computer and Information Sciences for his timely support during my research work.

My special thanks to my colleagues Abhimanyu Bar and Pandu Sakuntala for their invaluable support throughout the period of PhD work. And, I will never forget the evening tea break at the student canteen that was the favored moment of laughter, fun and discussion on research work with the supervisor and colleagues.

I would like to extend my sincere thanks to **Dr. Ni Peng**, **Dr. Xiao Zhang**, and **Dr. Yanyan Yang** for sharing the source code of PARA, FWARA, PIAR, IV-FS-FRS and AIFWAR algorithms, which are used in the comparative experimental study of the proposed approaches.

My sincere thanks to my friends Late Ravi Shanker Soni, Shailendra Sahu, Ishan Suryavanshi, Pooja Srivastava, Abhishek Alate, Surendra Mohan Raghav, Jitendra Khatwani, Renjith, Hima Mehta, Piyush Jain, Yogendra, Baby, Tripada, Gaurav, Bharat and my master's friends whose continuous support and encouragement is a great strength for my life's sojourn.

I also express my deep appreciation and thankfulness to my father, mother, brother and sister for extending their constant and generous support and belief in the completion of my PhD research work.

Last but not least, I give special thanks to myself for believing in me, doing all this hard work, and never quitting.

Anil Kumar

Abstract

In modern technologies, data has attracted significant attention from various fields due to its immense potential value, which can help in decision-making. However, data is being accumulated very fast and increasing amounts of data in size results in large-scale data. Processing and storing large-scale data can incur considerable memory costs and hamper the scalability of data mining algorithms.

One solution for tackling or scaling large-scale data is through granular computing (GrC) technology. GrC provides a conceptual framework in the domain of human-centric systems and computational intelligence. GrC involves the processing of complex information entities through information granules. Basically, GrC facilitates a higher-level view of data in terms of granules to tackle the problem much more efficiently. Integrating GrC and computational intelligence has become a desirable area for several researchers to develop efficient decision-making models for complex problems.

This thesis identifies fuzzy min-max neural network (FMNN) as a suitable technology for computing information granules due to their simplicity, effectiveness and robustness. FMNN was introduced by Patrick K. Simpson in 1992 as a supervised single-pass dynamic neural network classifier. FMNN creates n-dimensional hyperboxes to represent pattern spaces. FMNN has several salient properties that are suitable and adaptable for data mining tasks, such as online adaptation, non-linear separability, fast training time, and hard and soft decision-making ability. These hyperboxes as information granules conceptually capture the essence of the data concisely.

This research focuses on building hybrid soft computing models where FMNN is one of the components, and the hyperboxes are utilized in other components to achieve the advantages of granular computing. We investigate the applicability of FMNN induced hyperboxes in achieving a few data mining goals, particularly in building three models. The first part of the thesis is to build an efficient and enhanced FMNN classifier. The second part is to build a scalable feature subset selection approach using fuzzy rough sets. The last is to build an incremental feature subset selection approach using fuzzy rough sets. The present studies

considered these challenges in this thesis to be much more efficient with employing FMNN as a granular computing preprocessing strategy.

The first contribution of the thesis is towards overcoming limitations in FMNN. Several researchers have improved FMNN to overcome its limitations and minimize classification errors. However, these improved variants of FMNN still suffer from misclassification errors due to tampering with the non-ambiguous region and increased cost of training as the additional structure is added to the architecture of FMNN. An enhanced version of FMNN with kNN is proposed without altering the structure of FMNN and avoiding the contraction step. The proposed approach has the ability to handle decision-making in overlapped regions very efficiently.

The second contribution of the thesis is to increase the scalability of FRS approach. Fuzzy rough sets (FRS) theory is a hybridization of rough set theory (RST) and fuzzy sets that provides a framework for feature subset selection (also known as reduct computation). Traditional FRS approaches can't scale to large datasets due to the space complexity ($O(|U|^2|C|)$) where $|U|$ is the size of the object space and $|C|$ is the size of the attribute space. Several researchers have proposed the scalable FRS approach to deal with large datasets. However, these FRS approaches have been significantly scalable compared to traditional ones, but they still have not met much gain in computation time to compute reduct computation on large datasets. In this thesis, a novel scalable FRS-based reduct computation approach is proposed using FMNN as a preprocessing step that can enhance the scalability of FRS approaches. The proposed algorithm has achieved enhanced scalability to such an extent in large datasets where existing FRS algorithms are unable to compute. An extension to this work is also presented in the third contribution with the objective of further increasing the scalability and empirically arriving at recommendations about when to adopt these approaches.

Most FRS reduct computation approaches are restricted to batch processing; the entire data and its underlying structure are provided before training. When a new sample data arrives, the approach must recompute and reconstruct the model from scratch to compute a reduct. Several researchers have developed incremental reduct computation approaches to deal with dynamic datasets. But these incremental approaches are based on RST, not FRS. There are very few attempts made to investigate FRS-based incremental reduct computation. These incremental FRS approaches suffer in terms of their ability to scale to large datasets. In the fourth contribution, a novel scalable incremental FRS-based reduct computation approach is proposed using FMNN as a preprocessing step for dealing with dynamic datasets.

Comparative experimental analysis has been conducted for each contribution with existing state-of-the-art approaches over several benchmark datasets. Empirically, the results established that the proposed methods achieved higher scalability than compared approaches while achieving highly significant computational gain without compromising the performance of the classification models induced. In the future, we plan to further the scalability of our proposed methods through Apache Spark MapReduce distributive framework implementation that can deal with such voluminous datasets requiring memory beyond the availability in a single system.

Contents

List of Figures

List of Tables

Notations and Abbreviations

Γ: T-norm

$\mu_{DR_a}(x, y)$: A degree to which the objects x and y are similar for numerical attribute 'a'.

$\mu_{R_a}(x, y)$: A degree to which the objects x and y are dissimilar for numerical attribute 'a'.

θ: User defined parameter to constraint the size of hyperbox H

I^n: N-dimensional unit space

RST**:** Rough Set Theory

10-FCV: Ten-fold cross-validation technique

C^c: Set of categorical attributes

C^h: Set of hybrid conditional attributes

C^n: Set of numeric conditional attributes

{d}: A single decision attribute

C: Set of conditional attributes

CAverage: Average of the individual mean obtained by restricting to only those datasets in which all algorithms could be evaluated.

CDM-FMFRS: Crisp discernibility matrix based feature subset selection using FMNN

CDM: Crisp Discernibility Matrix

DISC: Discernibility relation

DM: Discernibility Matrix

DT: Decision System

NOTATIONS AND ABBREVIATIONS

FDM-FMFRS: Fuzzy discernibility matrix based feature subset selection using FMNN

FDM: Fuzzy Discernibility Matrix

FMNN: Fuzzy Min-Max Neural Network

FRS: Fuzzy Rough Sets

GrC: Granular Computing

H: Hyperbox

HBS: Set of hyperboxes

HDT: Hybrid Decision System

IDS: Interval-Valued Decision System

IND: Indiscernibility relation

IvFMFRS: Incremental FRS based feature subset selection using FMNN

kNN-FMNN: Integration of FMNN with kNN for enhancing classification performance

kNN: k-nearest neighbour

Neg: Negation

RST: Rough Set Theory

S: T-conorm

SAT(P): Satisfiability value of subset P for all the entries in the fuzzy discernibility matrix

SBE: Sequential backward elimination

SFS: Sequential forward selection

U: Set of the object space

Chapter 1

Introduction

1.1 Introduction

With the evolution of various modern technologies, the last two decades have witnessed rapid growth in both generating and collecting data. Information and technology have revolutionized large data collection [54, 134]. These datasets are collected from multiple sources of data such as enterprises, customer databases, public health, financial data etc. The structured form of the dataset for model construction is usually tabular in nature, where rows correspond to objects and columns correspond to features. These structured data have attracted significant attention from multiple applications due to their immense potential value/capability, which can help in decision-making challenges [6, 107]. The explosive increase in data requires new techniques that can transform the processed data into valuable knowledge. Consequently, data mining has evolved as an important research area to deal with data.

Data mining is an essential process for inferring the underlying structural patterns and knowledge from data and resulting it into valuable information [11, 109]. So, this valuable information is utilized by companies to uncover profitable patterns, increase their revenue amounts and decrease operational costs.

However, data is being accumulated very fast, and increasing amounts of data in size results in large-scale data. The processing capability of data mining techniques is critical under this periodical growth. Besides, storing large-scale data can result in a considerable memory cost and hamper the scalability of data mining algorithms. Even the data applicable for building applications is augmented with new data at different times or different circumstances (called dynamic data), which adds new difficulties for analysis.

One solution for tackling or scaling large scale data is through granular computing (GrC) technology [121, 127, 128]. GrC is an emerging computing paradigm for studying multidisciplinary information processing. It involves the processing of complex information entities through information granules. Information granule can be viewed as a composition of elements or objects of a universe drawn together by similarity, proximity, indistinguishability, or functionality [121]. In processing large-scale data, GrC establishes an effective role in providing an improved description of data which is cost-effective and computationally fast. Basically, GrC facilitates a higher-level view of data in terms of granules to tackle the problem much more efficiently. Integrating GrC and computational intelligence has become a desirable field for several researchers to develop efficient decision-making models for complex problems [107].

Here, we are building efficient solutions for data mining as data sizes are enormously increasing in scope, and are incrementally being acquired. Also dealing with data at the object level always has higher complexity in any facet of data mining. Hence, we acquire GrC technology which is good in both aspects, i.e., scalability and incremental adaptation. So, with the need for both fast computation and incremental adaptation, we have found a fine blend of all these aspects simultaneously satisfied in one technology called fuzzy min-max neural network (FMNN), introduced by Simpson in 1992 [102].

FMNN has several salient properties that are suitable and adaptable for data mining tasks, such as online adaptation, non-linear separability, fast training time, and hard and soft decision [102]. FMNN is a supervised single-pass dynamic neural network classifier to deal with pattern classification [102]. FMNN creates n-dimensional hyperbox fuzzy sets to represent pattern spaces, i.e., the union of fuzzy hyperboxes forms an individual pattern class [102]. Hyperboxes obtained from FMNN training can be viewed as information granules with characteristics of simple representation using minimum and maximum points and having a computationally efficient single-pass algorithm for constructing the same.

This thesis explores the applicability of FMNN in achieving a few data mining goals, particularly building an efficient and scalable classifier, a feature subset selection approach and an incremental feature subset selection approach. We have considered all these challenges in this thesis to be much more efficient with employing FMNN as a granular computing preprocessing strategy.

1.2 Motivation

With an extensive literature survey of FMNN and its extensions, we have arrived at the conclusion that FMNN and its extensions are primarily used for classification and clustering with applications in several real-world scenarios [1, 20, 85, 91]. Several researchers have utilized FMNN as a vehicle for a granular computing technique for computing information granules. Several researchers have introduced hybrid models in combination with FMNN to increase the ability of classification performance and computation power, such as FMNN with ant colony optimization [106], FMNN with particle swarm optimization [2], FMNN with decision tree [62], and FMNN with genetic algorithm [84] etc. We have observed that FMNN can significantly be useful in feature subset selection and incremental feature subset selection hitherto unexplored in the literature. This section provides the motivation and context for each problem considered in this thesis.

Enhancing Generalizability of FMNN:

In 1992, Simpson [102] proposed a supervised single-pass dynamic neural network classifier known as Fuzzy Min-Max Neural Network (FMNN) to deal with pattern classification. FMNN creates n-dimensional hyperbox fuzzy sets to represent pattern spaces. A fuzzy hyperbox is characterized by a minimum point, maximum point in n-dimensional pattern space [102]. FMNN learning is established by adjusting the min-max points of hyperboxes (information granules) using three steps, i.e., expansion criteria, overlap tests and contraction steps, to learn the pattern space [102].

FMNN is a robust and powerful learning model, though this model is still facing problems due to the contraction process, which may lead to gradation errors in classification. Contraction steps in FMNN tamper with the non-ambiguous region by modifying min-max points between hyperboxes in overlapped classes, inducing classification errors.

In the literature, several researchers have developed and improved traditional FMNN to overcome its limitations and minimize classification errors due to the contraction process [17, 52, 66, 81]. These variants in FMNN are in the direction of better representation of overlapping regions among hyperboxes, optimization and refinement of resulting hyperboxes and complementary with other soft computing models to enhance classification performance.

These improved variants of FMNN [17, 52, 66, 81] still suffer from misclassification errors due to tampering with the non-ambiguous region and the increased cost of training as the additional structure is added to the architecture of FMNN. These changes cause problems to the advantage of FMNN with simplified structure in terms of incremental adaptation etc. Hence this work investigates to identify the ways for ambiguity resolution in "FMNN without contraction" without altering the simple structure of FMNN.

Scalable Feature Subset Selection:

In the 1980s, Zdzisław I. Pawlak [77] introduced the concept of classical rough set theory (RST) as a mathematical tool, useful for feature subset selection in the information/decision systems [56, 78, 110, 123]. Primarily, RST is applicable to symbolic/categorical decision systems [77, 78, 123]. Application of RST to numeric decision systems will produce feature subsets with finer granularity. Hence, the induced rules from the selected features suffer from poor generalizability to test datasets. So, one of the solutions is to discretize the dataset beforehand and produce a new dataset with categorical values. But, any discretization process tends to cause a loss of information and result in classification error in pattern space [69].

Lately, Dubois et al. [23, 87] introduced fuzzy rough sets (FRS) theory which is a hybridization of rough sets and fuzzy sets that deals with both symbolic and real-valued conditional attributes. A subset of features selected using RST or FRS is named reduct, and the process is called reduct computation (feature subset selection).

However, traditional FRS approaches can't scale to large datasets due to the space complexity ($O(|U|^2|C|)$) where $|U|$ is the size of the object space and $|C|$ is the size of the attribute space. Several researchers have proposed the scalable FRS approach to deal with large datasets [15, 43, 72, 113, 138].

Even though these scalable FRS approaches have improved the scalability to some extent, the problem still requires a better solution for meeting today's emerging requirements for large data computation. Hence this work investigates in developing approaches for scalable FRS feature subset selection through FMNN-based granular computing.

Scalable Incremental Feature Subset Selection:

Most FRS reduct computation approaches are restricted to batch processing; the entire data and its underlying structure are provided prior to training at once. However, they are not designed to deal with dynamic datasets. When a new sample data arrives, the approach must recompute and reconstruct the model from scratch to compute a reduct.

In the literature, several researchers have explored how to process dynamic data through incremental learning methodologies that minimize the complexities of processing and storage. This idea has prompted several researchers to investigate the incremental perspective to feature selection in the framework of RST. These ideas have been investigated in various scenarios, such as the variation of feature set (adding and deleting features) and the sample set (adding and deleting objects), respectively. There have been a few studies on FRS based incremental feature selection algorithms under the variation of objects so far [73, 119, 120, 135].

Existing incremental FRS algorithms still suffer scalability issues due to object-based computations. Hence, this work investigates an approach for scalable incremental FRS feature subset selection through FMNN-based granular computing.

1.3 Problem Definition

This research focuses on building hybrid soft computing models where FMNN is one of the components, and the hyperboxes are utilized in other components to achieve advantages of granular computing. The objectives of the research are given below:

1. The first objective is to explore the methodology that combines the simple structure of FMNN with k-nearest neighbor (kNN) strategy for inducing a better classification model and incurring less computational time without resorting to modifying the structure of FMNN.

2. The second objective is to investigate a granular-computing based FRS reduct computation on achieving better scalability in reduct computation. The knowledge of FMNN can be utilized in decreasing the space complexity and the computational time required in FRS-based reduct computation.

3. The third objective is to investigate the incremental perspective of FRS approach using FMNN preprocessing to reduce the space complexity that can enhance the scalability of incremental FRS reduct computation.

The aforementioned objectives of this research can be summarized as follows: **The objective of this thesis is to evolve the hyperboxes to construct hybrid models for enhancing classification performance, formulating scalable approaches for reduct computation and incremental reduct computation for large decision systems.**

1.4 Major Contributions and Publications

The contributions of the thesis are elaborated towards the research motivation in which they are described in the preceding section. Each contribution and its corresponding publication are enumerated below:

Contribution 1 presents a hybridization of FMNN with kNN algorithm (kNN-FMNN) for performing the ability to handle decision-making in overlapped regions without altering the structure of FMNN. The work in this contribution has been published as given below.

- **Anil Kumar** and P. S. V. S. Sai Prasad, Hybridization of Fuzzy Min-Max Neural Networks with kNN for Enhanced Pattern Classification, In Advances in Computing and Data Sciences (ICACDS 2019), Pages 32-44, CCIS 1045, Springer 2019, ISBN 978-981-13-9939-8 **(Indexed in SCOPUS)**.

Contribution 2 proposes a novel FRS based feature subset selection approach (FDM-FMFRS) utilizing FMNN as a preprocessing step that can enhance the scalability of FRS approach. The work in this contribution has been published as given below.

- **Anil Kumar** and P. S. V. S. Sai Prasad, Scalable Fuzzy Rough Set Reduct Computation using Fuzzy Min–Max Neural Network Preprocessing, In IEEE Transactions on Fuzzy Systems, Vol. 28, Pages. 953-964, IEEE, May 2020. **(Indexed in SCI, SCOPUS)**.

Contribution 3 proposes an improvised FDM-FMFRS, named as CDM-FMFRS, in order to increase the scalability of feature subset selection. This work formulates a way to reduce space utilization in FDM-FMFRS that paves the way to increased scalability. The work in this contribution has been published as given below.

- **Anil Kumar** and P. S. V. S. Sai Prasad, Enhancing the Scalability of Fuzzy Rough Set Approximate Reduct Computation through Fuzzy Min-Max Neural Network and Crisp Discernibility Relation Formulation, In Engineering Applications of Artificial Intelligence, Vol. 110, Pages 104697, Elsevier, Apr 2022. **(Indexed in SCI, SCOPUS)**.

Contribution 4 proposes a scalable FRS-based incremental feature subset selection approach (IvFMFRS) using FMNN as a preprocessor step to deal with dynamic datasets. The work in this contribution has been published as given below:

- **Anil Kumar** and P. S. V. S. Sai Prasad, Incremental Fuzzy Rough Sets based Feature Subset Selection using Fuzzy Min-Max Neural Network Preprocessing. In International Journal of Approximate Reasoning, Vol. 139, Pages 69-87, Elsevier, Dec 2021. **(Indexed in SCI, SCOPUS)**

Additional Relevant Publications

Throughout my Doctoral research, I also contributed to the following collaborative publications. They are not acknowledged as contributions in the thesis.

- Abhimanyu Bar, **Anil Kumar**, P.S.V.S. Sai Prasad, Finding Optimal Rough Set Reduct with A* Search Algorithm, In Proceedings of Pattern Recognition and Machine Intelligence, PReMI 2019, Pages 317-327, LNCS 11941, Springer 2019, ISBN 978-3-030-34868-7. **(Indexed in SCOPUS)**

- Abhimanyu Bar, **Anil Kumar**, P.S.V.S. Sai Prasad. Coarsest granularity-based optimal reduct using A* search, In Granular Computing, Vol. 8, Pages 45-66, Springer, January 2023. **(Indexed in ESCI, SCOPUS)**

1.5 Organization of the Thesis

The thesis has been structured into seven chapters based on methods.

Chapter 1 presents the introduction part of the thesis. It reviews the usefulness of granular computing in data mining and its usefulness in the feature subset selection approach. The chapter also enumerates research motivation, objectives, and the thesis's contributions and organization.

Chapter 2 presents an introduction to the concept of granular computing and a brief overview of FMNN as a granular computing method which is helpful in enhanced understanding of the contributions.

Chapter 3 discusses a literature review of variants of FMNN and their limitations. Based on the study of related literature, a proposed enhanced version of FMNN model, in terms of classification performance and computational time, is presented. The algorithm developed in this chapter is kNN-FMNN. Comparative experimental analysis of kNN-FMNN with state-of-the-art approaches is presented on benchmark datasets. The contribution of this chapter is published in the proceedings of ICACDS-2019.

Chapter 4 introduces the basics of classical rough sets and fuzzy rough sets with discernibility matrix construction and attribute reduction (reduct computation) process. Also, it discusses the literature review of existing scalable FRS reduct computation approaches and their limitations. A proposed scalable FRS reduct computation using FMNN as a preprocessing step is introduced. The algorithm developed in this chapter is FDM-FMFRS. Comparative experimental analysis of FDM-FMFRS with state-of-the-art approaches is given on benchmark datasets. The contribution of this chapter is published in IEEE Transactions on Fuzzy Systems.

Chapter 5 presents the inherent possible extensions of a methodology developed in Chapter 4 in terms of enhancing further scalability. The algorithm developed in this chapter is CDM-FMFRS. Comparative experimental analysis of CDM-FMFRS with FDM-FMFRS and compared algorithms are given on benchmark datasets. The contribution of this chapter is published in Engineering Applications of Artificial Intelligence.

Chapter 6 contains the literature review of existing FRS incremental reduct computation approaches and their limitations. A proposed incremental reduct computation in FRS with FMNN preprocessing step is presented. The algorithm developed in this chapter is IvFMFRS. Comparative experimental analysis of IvFMFRS with state-of-the-art incremental approaches is reported. The contribution of this chapter is published in International Journal of Approximate Reasoning.

The thesis concludes with **Chapter 7**, which summarizes the research contributions and presents directions for future work.

Chapter 2

Granular Computing using Fuzzy Min-Max Neural Network

This chapter addresses the basic concepts that are used throughout this thesis to understand the proposed works. Section 2.1 describes an overview of granular computing. Section 2.2 presents one of the granular computing models known as fuzzy min-max neural network and its architecture and classification process.

2.1 Granular Computing

When humans observe a set of unknown characters, images, or objects that are not familiar to them, they tend to group them by their similarity, shape, or size and form an abstract view of those specific things for further decision-making. This renders human cognition involving several levels of granularity (i.e., abstraction) to understand the newly acquired information and make our ensuing cognitive process more effective [88].

Granular computing (GrC, in short) is an emerging computing paradigm in the field of studying multidisciplinary information processing and provides the conceptual framework in the domain of human-centric systems and computational intelligence. GrC is subjective for understanding how humans granulate concepts or features and execute rational decisions in uncertain and imprecise environments. GrC can obtain different aspects of knowledge, as well as enhance the understanding of the underlying knowledge structure.

The word granularity has been first used in Loft A. Zadeh's 1979 paper, "Fuzzy Sets and Information Granularity"[125]. Granulation is a process that constructs or decomposes a universe into granules. Lofti Zadeh's keynote speech [129] on granulation states:

"Information granulation involves partitioning a class of objects (points) into granules, with a granule being a clump of objects (points) which are drawn together by indistinguishability, similarity, or functionality."

GrC is thus important in human problem-solving and has a very significant impact on the design and implementation of intelligence systems. GrC involves the processing of complex information entities through "information granules". A granule can be viewed as a composition of elements or objects of a universe as they can be drawn together by similarity, proximity, indistinguishability or functionality. Information granule is a primitive concept in granular computing. Basically, GrC is all about representing, constructing and processing information granules.

Fuzzy min-max neural network (FMNN) is an emerging soft computing paradigm of granular computing [102]. FMNN creates n-dimensional hyperbox fuzzy sets to represent pattern spaces, i.e., the union of fuzzy hyperboxes forms an individual pattern class. This thesis identifies FMNN as a suitable technology for computing information granules.

Our research focuses on building hybrid soft computing models where FMNN is one of the components, and the hyperboxes are utilized in other components to achieve the advantages of granular computing.

2.2 Overview of Fuzzy Min-Max Neural Network

In 1992, Simpson [102] proposed a single-pass dynamic neural network structure to deal with pattern classification known as fuzzy min-max neural network (FMNN). This approach presented in [101] as an extension of earlier work [100] to learn pattern classes. There are several salient properties of FMNN for using it as a classification model, which are briefly described as follows:

1. **Online adaptation:** FMNN has the ability to learn new classes and refine existing classes with new input patterns over time without eliminating old or previous classes or retrain the learning model. Hence, FMNN provides an appropriate solution for "Stability-Plasticity Dilemma" [8] problem.

2. **Non-linear separability:** FMNN classifier can allow various classes to build non-linear separable decision regions for classification that separate classes of any shape and size.

3. **Overlapping classes:** FMNN has the ability to minimize the misclassification of patterns in all overlapping classes on decision boundaries.

4. **Non-parametric classification:** FMNN classifier doesn't depend on any prior knowledge of the underlying data distribution, thereby is able to provide reliable decision

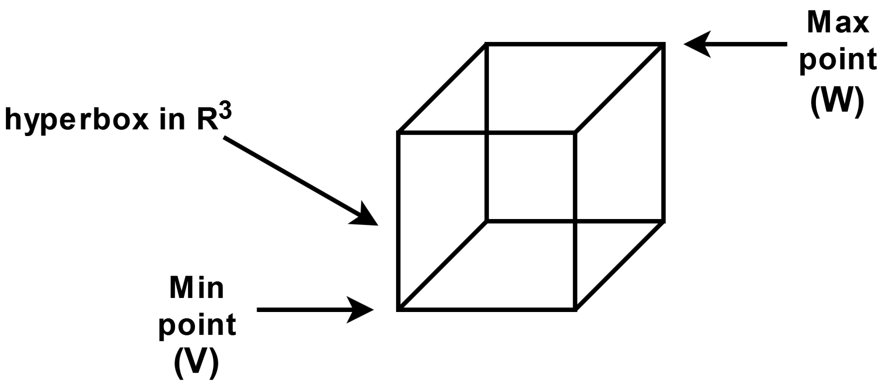


Figure 2.1: Hyperbox in 3-D

boundaries.

5. **Hard and soft decision:** FMNN has the ability to provide both hard and soft classification decisions. For hard decisions, the decision regarding input test data is crisp, i.e., either 0 or 1. For soft decisions, a classifier describes the degree to which input test data fits in each class.

6. **Training time:** FMNN is a single-pass algorithm. It learns very fast in comparison to other non-linear classification algorithms that require huge computational time to learn decision boundaries. These other non-linear algorithms require many passes through the data to achieve optimal objective function to learn decision boundaries as in the case of back-propagation algorithm [90].

FMNN is a supervised learning neural network that uses n-dimensional hyperbox fuzzy sets to represent pattern spaces [102]. Each hyperbox restricts a subregion defined by pairs of minimum point (V) and maximum point (W), and it characterized by a fuzzy membership function. The hyperbox having min and max points in 3-dimensional space is depicted in Fig. 2.1. Basically, the author presents hyperbox as fuzzy sets due to their corresponding membership function that allows them to create fuzzy sets in n-dimensional space.

This membership function of the hyperbox describes the degree of pattern fitting within the restricted region. The maximum size of hyperbox along each dimension is restricted by theta (θ), which is user-defined parameter with range of $[0, 1]$ ($0 < \theta \leq 1$). Therefore, the pattern space will be constrained into the n-dimensional unit cube I^n. Given a numeric decision system, all the numeric attributes are scaled into $[0, 1]$ before applying FMNN. Each hyperbox H_j is defined as:

$$H_j = \{X_h, V_j, W_j, Memb_j(X_h)\} \quad \forall\ X_h \in I^n \tag{2.1}$$

where $X_h = (x_{h1}, x_{h2}, ..., x_{hn})$ is an input pattern in n-dimensional space, and $V_j = (v_{j1}, v_{j2}, ..., v_{jn})$ and $W_j = (w_{j1}, w_{j2}, ..., w_{jn})$ are the corresponding minimum point and maximum point for hyperbox H_j. $Memb_j(X_h)$ is the fuzzy membership function that describes the membership value of the input pattern X_h w.r.t particular H_j hyperbox and defined as:

$$Memb_j(X_h) = \frac{1}{2n}\sum_{i=1}^{n}[\ max(0,\ 1 - max\,(\ 0,\ \gamma.min\,(\ 1, x_{hi} - w_{ji})))$$
$$+max(\ 0, 1 - max\,(\ 0, \gamma.min\,(\ 1, v_{ji} - x_{hi})))] \tag{2.2}$$

where, γ provides the sensitive parameter described to the pace of decrease of the fuzzy membership values, and $0 \leq Memb_j(X_h) \leq 1$. The fuzzy membership function computes on a dimension by dimension to measure the degree how far each component is lesser (greater) than the minimum (maximum) point value along with each dimension that falls outside the min-max bounds of the hyperbox. As the membership approaches one, the point should be more contained by the hyperbox, with the value one representing complete hyperbox containment. The membership function (defined in Eqn. (2.2)) is the sum of two components, first the average amount of max point violation and the average amount of min point violations.

The aggregation of hyperbox fuzzy sets creates the decision boundaries that separate classes. So, aggregation of fuzzy set that classifies the k^{th} pattern class (C_k) is defined in Eqn. (2.3).

$$C_k = \bigcup_{j \in K} H_j \tag{2.3}$$

where, K is the index set of the hyperboxes associated with the class k. An example of aggregation of hyperboxes separated by decision boundary in 2-D is depicted in Fig. 2.2.

2.2.1 Architecture of FMNN Classifier

The topology of FMNN classifier is a three-layer feed-forward neural mechanism, as shown in Fig. 2.3. The first layer (F_A) is an input layer for input patterns in n-dimensional space; the second is a hidden layer where each node represents a hyperbox fuzzy set; and the third is an output layer where each node represents a decision class. Each node in the input layer (F_A) is connected with every node in the hidden layer (F_H) with two connection weights minimum (stored in V matrix) point and maximum (stored in W matrix) point. A fuzzy membership function is considered as (F_H) a transfer function, defined in Eqn. (2.2). The connection between (F_H) and (F_C) nodes is binary-valued and stored in the matrix U, as defined in

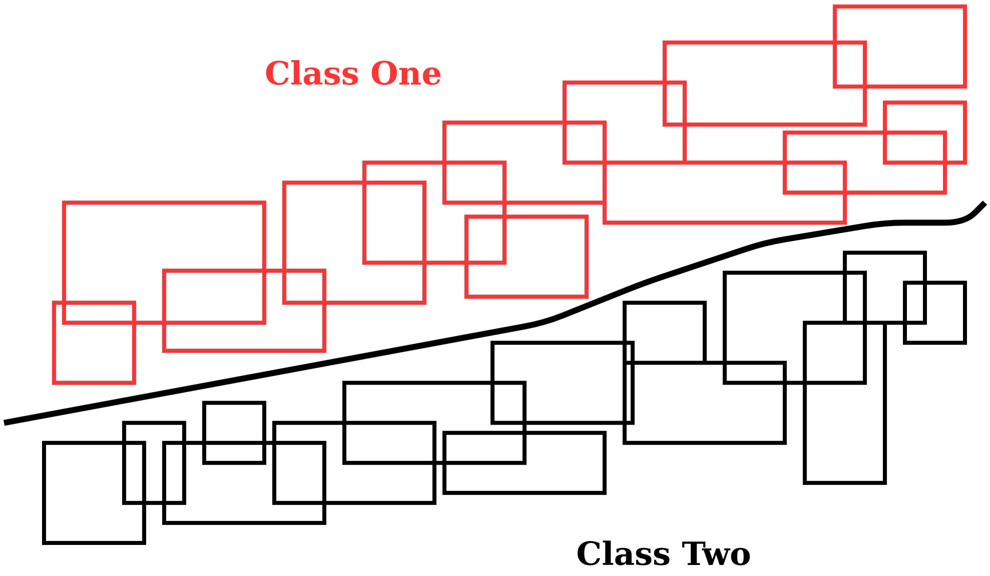


Figure 2.2: An Example of FMNN Hyperboxes Placed Along Boundary of a Two-Class Problem

Eqn. (2.4).

$$u_{jk} = \begin{cases} 1 & \text{If } H_j \; is \; a \; hyperbox \; for \; class \; C_k \\ 0 & otherwise, \end{cases} \tag{2.4}$$

The output of each (F_C) node shows the membership degree to which the input pattern belongs to a decision class. The transfer function for each of (F_C) nodes in the output layer performs the fuzzy union of corresponding hyperbox fuzzy set values, as described in Eqn. (2.5).

$$C_k = \max_{j=1}^{m} H_j u_{jk} \tag{2.5}$$

2.2.2 Classification Learning in FMNN

FMNN learning process is performed by creating and adjusting hyperboxes in n-dimensions space for all decision classes. The learning process begins with an input pattern $\{X_h, C_h\}$ that enters the network. We compute the membership of X_h into all hyperboxes of decision class C_h. If any of the membership values is one, in other words, the input pattern absolutely belongs to one of the existing same class hyperbox, then no further training is required. Otherwise, the network tries to find the same class hyperbox that can expand to accommodate the input pattern through the expansion process (if needed). If the hyperbox cannot meet the expansion criteria to include the input pattern, then a new hyperbox is created and added

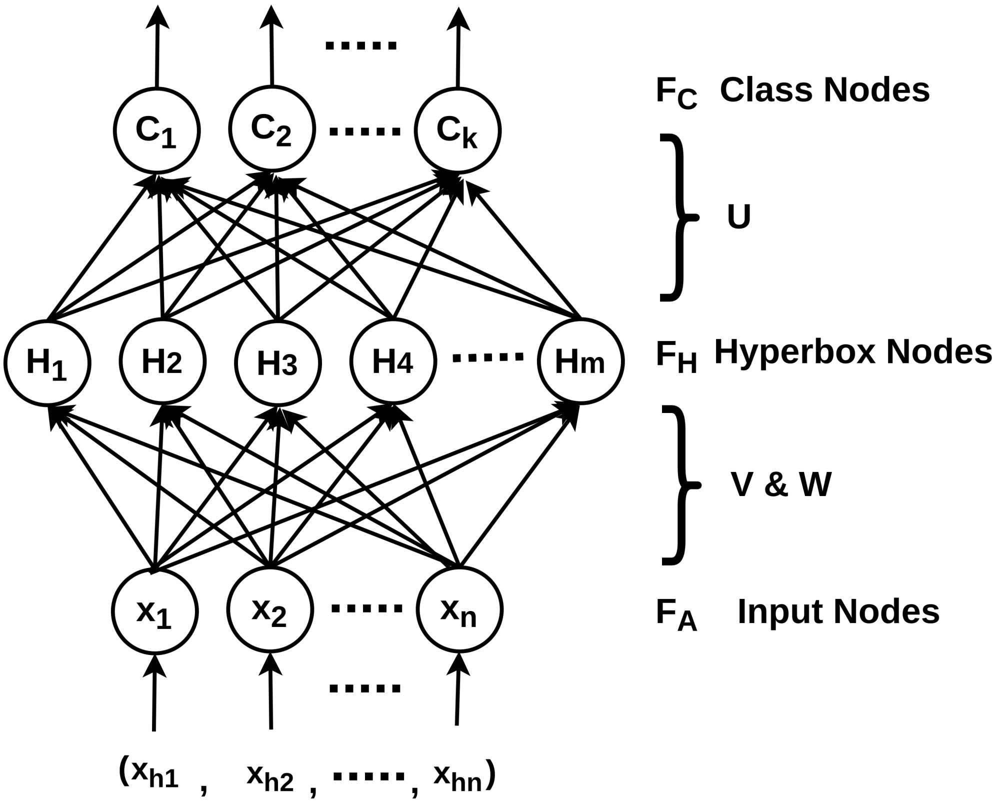


Figure 2.3: Three Layer Neural Network Architecture of FMNN

to the network. If a hyperbox expansion has happened, there is a chance of overlapping among similar or different class existing hyperboxes. Usually, the overlap between hyperboxes representing the same class is not a problem. But, overlap among hyperboxes from other classes is important and needs to be eliminated using a contraction process.

FMNN training involves three stages for acquiring knowledge: *Hyperbox Expansion process, Overlap test and Contraction process.*

Hyperbox Expansion: Given an input pattern $\{X_h, C_h\}$, the network identifies a winning hyperbox with the highest degree of membership value and represents the same decision class as C_h for expansion. For the expansion, H_j hyperbox must be bound by the expansion criteria constraint given in Eqn. (2.6), to include an input pattern X_h.

$$\sum_{i=1}^{n} (max(w_{ji}, x_{hi}) - min(v_{ji}, x_{hi})) \leq n\theta \tag{2.6}$$

where, the range of user-defined parameter (θ) in Eqn. (2.6) is within the range of ($0 < \theta \leq 1$) and controls the maximum size of a hyperbox.

If the expansion criterion, given in Eqn. (2.6), is satisfied between the input pattern X_h and hyperbox H_j, then the minimum and maximum points of hyperbox H_j are altered to accommodate the input pattern X_h using Eqn. (2.7) and Eqn. (2.8).

$$v_{ji}^{new} = min(v_{ji}^{old}, x_{hi}) \quad \forall i = 1, 2, 3, \ldots, n. \tag{2.7}$$

$$w_{ji}^{new} = max(w_{ji}^{old}, x_{hi}) \quad \forall i = 1, 2, 3, \ldots, n. \tag{2.8}$$

If the existing hyperbox H_j can not be expanded using Eqn. (2.6), then a new point hyperbox is created to contain X_h, whose min and max points are set to X_h.

Overlap Test: After the expansion process, there is a chance that H_j leads to overlapping with adjacent hyperboxes representing different classes. The overlap test is to determine any chance of overlapping between hyperboxes through dimension by dimension comparison. Two hyperboxes don't overlap as long as there is at least one dimension that is not overlapping. If overlap existed between two hyperboxes, at least one of the following four cases is satisfied in each dimension. Suppose both H_j and H_k hyperboxes represent different classes are being examined for possible overlap. Assuming, $\delta^{old} = 1$, four test cases and their corresponding minimum overlap value for i^{th} dimensions are as follows:

$$\begin{aligned}
&case\ 1: v_{ji} < v_{ki} < w_{ji} < w_{ki}\\
&\qquad \delta^{new} = min(w_{ji} - v_{ki}, \delta^{old})\\
&case\ 2: v_{ki} < v_{ji} < w_{ki} < w_{ji}\\
&\qquad \delta^{new} = min(w_{ki} - v_{ji}, \delta^{old})\\
&case\ 3: v_{ji} < v_{ki} < w_{ki} < w_{ji}\\
&\qquad \delta^{new} = min(min(w_{ki} - v_{ji}, w_{ji} - v_{ki}), \delta^{old})\\
&case\ 4: v_{ki} < v_{ji} < w_{ji} < w_{ki}\\
&\qquad \delta^{new} = min(min(w_{ki} - v_{ji}, w_{ji} - v_{ki}), \delta^{old})
\end{aligned} \tag{2.9}$$

If $\delta^{old} - \delta^{new} > 0$, then $\Delta = i$ and $\delta^{old} = \delta^{new}$, signifying that there is overlap in Δ^{th} dimension, and the next overlapping testing will continue for the next dimension. If none of the conditions is satisfied, then there is no overlapping in the i^{th} dimension and hence no overlapping among hyperboxes. In which case, overlapping checking between hyperboxes will not proceed, and Δ is set to indicate that the contraction process is not required, i.e., $\Delta = -1$.

One can say that hyperboxes are not overlapped means that at least in one dimension, there is no overlap. Hence, there should be overlapping in each dimension to say that both hyperboxes share boundary regions.

Contraction Process: If $(\Delta > 0)$, then an overlap existed between hyperboxes (H_j and H_k) on Δ^{th} dimension is adjusted using the contraction process. Because the smallest dimension minimally affects the state of hyperboxes and keeps hyperbox size as large as possible, that delivers a more robust pattern classification. For contraction, four cases are examined for adjustment between hyperboxes, as given below:

$$\begin{aligned}
&case\ 1: v_{j\Delta} < v_{k\Delta} < w_{j\Delta} < w_{k\Delta}\\
&\quad w_{j\Delta}^{new} = v_{k\Delta}^{new} = \frac{w_{j\Delta}^{old} + v_{k\Delta}^{old}}{2}\\
&case\ 2: v_{k\Delta} < v_{j\Delta} < w_{k\Delta} < w_{j\Delta}\\
&\quad w_{k\Delta}^{new} = v_{j\Delta}^{new} = \frac{w_{k\Delta}^{old} + v_{j\Delta}^{old}}{2}\\
&case\ 3: v_{j\Delta} < v_{k\Delta} < w_{k\Delta} < w_{j\Delta}\ and\ (w_{k\Delta} - v_{j\Delta}) < (w_{j\Delta} - v_{k\Delta})\\
&\quad v_{j\Delta}^{new} = w_{k\Delta}^{old}\\
&case\ 4: v_{j\Delta} < v_{k\Delta} < w_{k\Delta} < w_{j\Delta}\ and\ (w_{k\Delta} - v_{j\Delta}) > (w_{j\Delta} - v_{k\Delta})\\
&\quad w_{j\Delta}^{new} = v_{k\Delta}^{old}\\
&case\ 5: v_{k\Delta} < v_{j\Delta} < w_{j\Delta} < w_{k\Delta}\ and\ (w_{k\Delta} - v_{j\Delta}) < (w_{j\Delta} - v_{k\Delta})\\
&\quad w_{k\Delta}^{new} = v_{j\Delta}^{old}\\
&case\ 6: v_{k\Delta} < v_{j\Delta} < w_{j\Delta} < w_{k\Delta}\ and\ (w_{k\Delta} - v_{j\Delta}) > (w_{j\Delta} - v_{k\Delta})\\
&\quad v_{k\Delta}^{new} = w_{j\Delta}^{old}
\end{aligned} \tag{2.10}$$

The overlap and contraction step between hyperboxes H_1 and H_2 of different classes are illustrated in Fig. 2.4. The shaded region showed the overlapping part between hyperboxes. So, the overlap test finds the minimum overlap region along the x-axis dimension. Then, the contraction steps alter their min and max points between the hyperboxes along the selected dimension to eliminate ambiguity, as shown in bold outline in Fig. 2.4.

It can be observed based on the training algorithm of FMNN and being a single iteration-based algorithm, FMNN is an ordered-dependent algorithm. By this, we mean that the formation of hyperboxes varies in case the same training objects are presented to FMNN in a different order.

In testing phase of FMNN classifier, for a given test pattern X, the fuzzy membership of

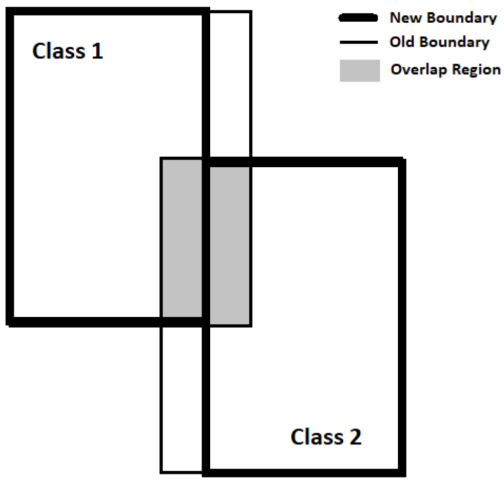


Figure 2.4: Overlap and Contraction Process Between Hyperboxes

X is computed with respect to all the hyperboxes. The test pattern X is classified as the decision class corresponding to the hyperbox achieving the highest fuzzy membership or full membership.

These placings and adjustments of hyperboxes create a granular structure of pattern in pattern space which is useful for pattern classification. This method also establishes several salient learning features like online learning, non-linear separability and non-parametric classification, thus, making FMNN more flexible. The main advantage of the FMNN is that it has the potential to learn approximate decision concepts through single pass training. The unique blend of single epoch learning combined with adaptability to incremental learning has made the FMNN suitable for current scenarios of building intelligent systems in an online environment.

Hyperboxes obtained from FMNN training can be viewed as information granules with characteristics of simple representation using minimum and maximum points and having a computationally efficient single-pass algorithm for constructing the same. The primary objective of our research work is to explore the potential possibility of utilizing information granules in the form of hyperboxes and formulating algorithms for granular computing using hyperboxes in solving the standard problems of data mining and machine learning. Our research focuses on building hybrid soft computing models where FMNN is one of the components and the hyperboxes are utilized in other components to achieve advantages of granular computing.

In this thesis, we formulate, design, and develop granular computing-based solutions using FMNN induced hyperboxes for the following three problems:

1. Efficient classifier model construction for overcoming "contraction step" induced problems.
2. Feature subset selection using fuzzy rough set theory.
3. Incremental feature subset selection using fuzzy rough set theory.

The rest of the chapters explained each of the contributions.

Chapter 3

Enhancement of Fuzzy Min-Max Neural Network for Classification

Fuzzy min-max neural network (FMNN) is a single-pass dynamic neural network classifier to deal with pattern classification. Indeed, the theory has been performing remarkably with further extensions and modifications to enhance the pattern classification in recent years. However, despite these modifications and extensions, these variants result in an increase in the computational cost due to additional constructs in the architecture of FMNN and loss of information owing to the contraction step.

This chapter highlights the related issues associated with FMNN methodology and its variants and provides a solution that can enhance the pattern classification and incur less computational time.

The rest of the chapter is designed as follows: Section 3.1 briefly introduces the literature survey of variants of FMNN and their disadvantages. Section 3.2 presents the motivation behind the proposed algorithm. Section 3.3 briefly describes the functioning of the proposed algorithm kNN-FMNN. Section 3.4 describes the complexity analysis of proposed algorithm kNN-FMNN. Section 3.5 reports a series of experiments and comparative analysis of kNN-FMNN with state-of-the-art approaches.

3.1 Literature Review FMNN Variants

In 1965, Zadeh [126] introduced fuzzy sets as an extension of the classical sets to describe and manipulate data that are not precise. Fuzzy logic is a generalization of fuzzy sets in which a concept is characterized by a degree of membership ranging between zero and one. Fuzzy logic aims at creating approximate human reasoning that is helpful for cognitive decision-making. Several hybrid systems have been developed by researchers with fuzzy sets combining other soft computing models such as artificial neural networks, expert systems and genetic algorithms etc [27, 99, 139, 141].

A hybrid system like the artificial neural network with fuzzy logic has proven its effectiveness in real-world problems [27]. The main advantage of artificial neural systems is their adaptability, making models good at understanding patterns but not enough to explain how to reach their soft decisions. So, fuzzy logic systems aid the neural network in the enhancement of interpretability.

In 1992, Simpson [102] proposed a supervised single-pass dynamic neural network classifier known as Fuzzy Min-Max Neural Network (FMNN) to deal with pattern classification using fuzzy sets as pattern classes. FMNN employs n-dimensional hyperbox fuzzy sets to represent pattern spaces, i.e., the union of fuzzy hyperboxes forms an individual pattern class. A fuzzy hyperbox is represented as a region in n-dimensional pattern space and characterized by minimum point, maximum point and fuzzy membership function [102]. FMNN learning is established by adjusting the min-max points of hyperboxes (information granules) to acquire or learn knowledge of the pattern space. This way FMNN exhibits a non-linear separability property of finding decision boundaries across decision classes. Complete details of FMNN and its procedure for training and testing details are explained in Chapter 2.

FMNN has been applied successfully in different applications such as fault detection, lung cancer, medical data analysis, image processing, video sequence segmentation and text classification etc [1, 20, 67, 85, 86, 91, 93, 94, 142]. For example, in image segmentation, instead of processing individual color pixels, a group of pixels (granules) can be processed efficiently using GRFMNN [67]. GRFMNN model is used to build up granules through training min-max values of the pixels in each grid. These granules are then used for classification. This way significantly reduces the computational costs required to process individual pixels. Similarly, GRFMNN is used to eliminate shadows from color images and also reduces the dependability of the existing computer vision approach. There are many examples of FMNN that provides practical solutions in real-world applications.

Although FMNN is a robust and powerful learning model, this model is still facing problems due to the contraction process, which may lead to gradation errors in classification. Contraction steps in FMNN lead to tempering with the non-ambiguous region by modifying min-max points between hyperboxes in overlapped classes, which can induce classification errors.

These issues have motivated several researchers to develop and improve the FMNN to overcome its limitation and minimize the misclassification error due to the contraction process. These variants in FMNN are in the direction of better representation of overlapping regions among hyperboxes, optimization and refinement of resulting hyperboxes and complementary with other soft computing models to enhance classification performance.

Our studies can be viewed as a literature review of existing FMNN approaches in two different categories (with contraction and without contraction) that include various FMNN variants, as depicted in Fig. 3.1. The first category is to retain traditional FMNN learning stages (expansion, overlap, and contraction steps) along with modifications and enhancements. The second category highlights the FMNN variants that eliminate the contraction procedure.

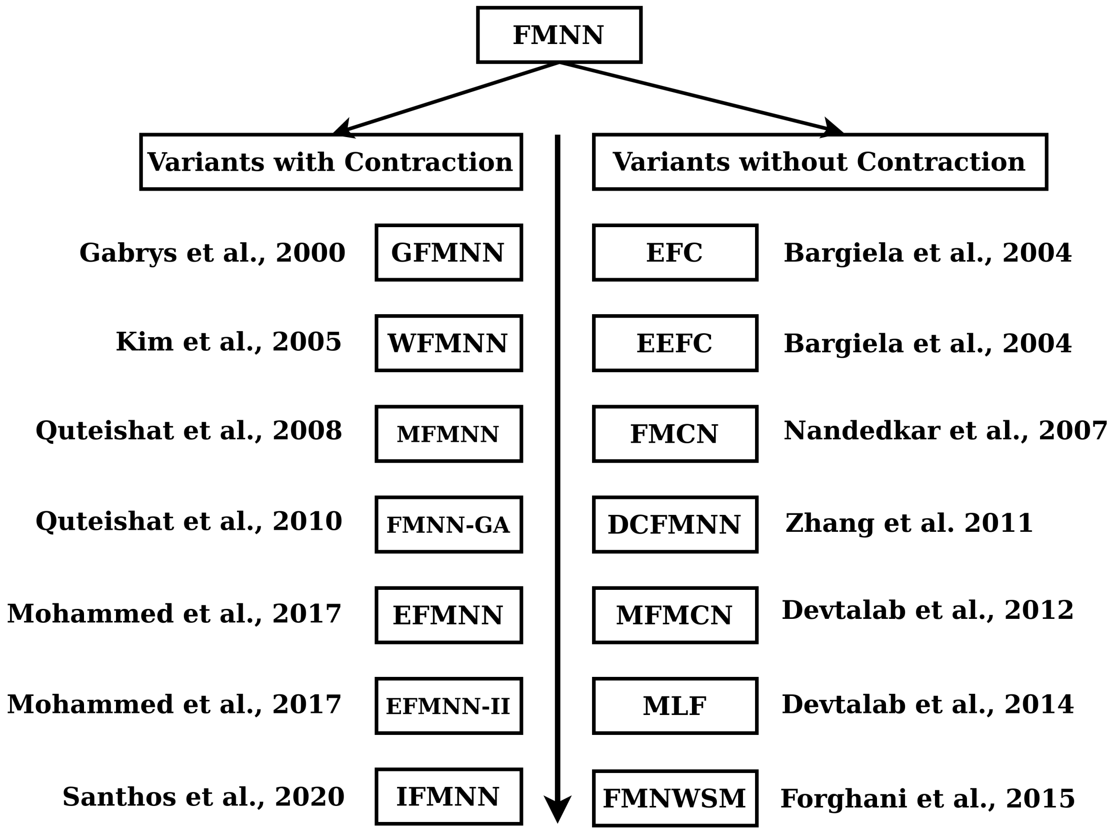


Figure 3.1: Variants of FMNN With and Without Contraction Process

3.1.1 FMNN Variants with Contraction

In 2000, Gabrys et al. [28] proposed a generalization and extension of FMNN, called a General Fuzzy Min-Max Neural Network (GFMNN) to enhance FMNN classification's effectiveness by addressing a few issues in using traditional FMNN. These issues are related to fuzzy membership function and hyperbox expansion criterion. GFMNN appears to work on both supervised and unsupervised learning within a framework, while traditional FMNN presents two different approaches. The authors also propose a new membership function describing the degree to which an input pattern belongs within the hyperbox and a new expansion criterion to expand the hyperbox to cover the input pattern. Comparatively, GFMNN achieves better pattern classifiability by generating fewer hyperboxes than FMNN.

In 2005, Kim et al. [49] proposed an extension of FMNN, called a Weighted Fuzzy Min-Max Neural Network (WFMNN) that considers weights into account. The author gives the importance of each feature in each hyperbox using weights. This weight value is assigned to a feature based on the frequency of occurrence of patterns against other features of the same hyperbox. The authors also present a new fuzzy membership function by considering the weight factor which encourage to exploit the importance of features. The proposed model compensates for the distortion and noise of the hyperbox during expansion and contraction steps by employing feature distribution information. WFMNN was successfully applied in the fields of feature extraction [51] and face detection applications [50].

In 2008, Quteishat et al. [86] proposed a modification of FMNN as MFMNN in an endeavor to increase the classification performance of FMNN. MFMNN strategy focuses on the scenario when a few large hyperboxes are created, i.e., expansion parameter (θ) is large. Authors incorporate a confidence factor-based pruning strategy into FMNN to remove low confidence factor hyperboxes after training the FMNN network. Also, they include the Euclidean distance along with fuzzy membership function in the FMNN network to predict the test pattern class, especially when the θ is large. The winning hyperbox is the one obtaining the shortest Euclidean distance from test pattern to the centroid of the hyperbox.

In 2010, Quteishat et al. [84] proposed an extension of MFMNN, called a MFMNN with genetic algorithm (GA)-based rule extractor (MFMNN-GA) for enhancing the classification performance. The first stage of MFMNN-GA is to generate hyperboxes through the base model (MFMNN) and then prune hyperboxes using a confidence factor to decrease the model's complexity. The idea of using the confidence factor is for identifying the frequent occurring hyperboxes. The pruning method removes low confidence factor hyperboxes to minimize the network complexity. The second stage is to apply 'don't care' strategy by GA-rule extractor to reduce the number of features in the extracted rule and improve the classification performance.

In 2017, Mohammed et al. [63] presented an improved FMNN, called an Enhanced Fuzzy Min-Max Neural Network (EFMNN), to address the limitation in the learning process of FMNN and improve the performance of classification. These limitations address overlapping rules and contraction rules to remove the overlapped region between hyperboxes. Authors extend standard overlapping steps with new ones to manage all possible overlapped regions between hyperboxes missing in the earlier one. Besides, a new contraction step is also provided to resolve all possible overlapping cases.

In 2017, Mohammed et al. [64] introduced an enhanced version of FMNN, called a FMNN with a K-nearest hyperbox expansion rule (KnFMNN) to improve the classification. Authors associated a new hyperbox expansion rule using k-NN strategy to reduce FMNN network complexity. In KnFMNN, a set of k hyperboxes is selected to cover input patterns, i.e., if one hyperbox is not satisfied with the expansion criteria, then the next hyperbox is considered for expansion till the set goes empty. This has resulted in the model being more generic and creating fewer hyperboxes, thus increasing the classification performance.

In 2017, Mohammed et al. [65] presented an extension of EFMNN as EFMNN-II by incorporating two strategies: k-nearest hyperbox expansion rule and pruning strategy. k-nearest hyperbox expansion rule is employed to select the winning hyperbox, and the pruning process is formulated to eliminate less efficient hyperboxes. This way increases EFMNN performance in terms of classification and network complexity.

In 2020, Santhos et al. [52] presented an enhanced version of FMNN, called Improved FMNN (IFMNN), to increase classification performance. The authors employ k-nearest hyperbox expansion rule along with the perimeter of hyperbox to choose a winning hyperbox for expansion. And a weighted procedure based on the perimeter is proposed to check the expandability of the selected hyperbox. Also, a set of contraction rules based on FMNN and EFMNN are altered using a perimeter of a given hyperbox to balance the overlapping regions. IFMNN is a refinement of FMNN, k-FMNN and EFMNN approaches.

Although these proposed approaches (having contraction steps) are an enhanced and improved version of the original FMNN to reduce classification error as aforementioned, these FMNN variants still suffer the data distortion and gradation error, which may result in a classification error. These variants still employ the same contraction process as FMNN with few improved versions that tempered the acquired knowledge in the non-ambiguous region, causing gradation error in classification.

3.1.2 FMNN Variants without Contraction

Many researchers have achieved an innovative way to exclude the contraction process in FMNN to retain overlapping information for better pattern classification.

In 2004, Bargiela et al. [4] proposed an improved FMNN classifier, known as inclusion/exclusion fuzzy hyperbox classifier (EFC). It provides a new learning methodology to deal with the overlapping region problem in FMNN by dropping the contraction process. EFC considers two types of hyperboxes named as inclusion and exclusion hyperboxes. Inclusion hyperboxes can contain input patterns belonging to the same decision class. The exclusion hyperboxes include input patterns that fall in the overlap region of different class hyperboxes. Using exclusion hyperboxes reduces FMNN three-step learning process (expansion, overlap test and contraction) into two steps process (expansion and overlap test). However, this method resulted in a reduction in misclassification owing to the discarding of exclusion hyperboxes.

In 2004, Bargiela et al. [89] proposed an extension of EFC, called a Adaptive Inclusion/Exclusion Fuzzy Hyperbox Classifier (EEFC) to improve classification performance. Like EFC, the authors consider two kinds of hyperboxes named inclusion and exclusion hyperboxes but use a modified expansion step. In FMNN, the maximum size of the hyperbox is to be fixed in advance. In this paper, the authors induce the adaptive nature of the expansion parameter of all hyperboxes, which means no parameter is fixed in advance. This way, the overlap regions don't become too large, which is possible in EFC algorithm.

In 2007, Nandedkar et al. [66] introduced a novel FMNN classifier, called a Fuzzy Min-Max Neural Network Classifier with Compensatory Neurons Architecture (FMCN). FMCN incorporates two additional types of compensatory neuron architecture along with classifying neurons (CLNs, which represent a pure hyperbox) in the original FMNN architecture named as containment compensation neurons (CCNs) and overlapped compensation neurons (OCNs). The idea of FMCN is to protect the min-max points of the overlapped region between hyperboxes using compensatory neurons to address overlapped regions. CCNs represent an overlap region (containment region), where the hyperbox is entirely and partially encloses another hyperbox belonging to a different class. OCNs address the overlap region between hyperboxes of distinct classes, where a new hyperbox is created to represent the overlap region's size. A new fuzzy membership function is also presented for compensatory neurons. This method can protect the min-max points of the overlap region to enhance the learning algorithm as this information is highly significant for pattern classification.

In 2007, Zhang et al. [131] proposed a new approach, called a Data Core Based Fuzzy Neural Network (DCFMN), to overcome the limitation of FMCN with the help of the geometrical center and data core of hyperbox. DCFMN also has the benefit of handling noisy data. Like FMCN, DCFMN also contains a compensatory neuron and classifying neurons (CLNs, representing a pure hyperbox). Compensatory neurons address all form of overlapping region problems among hyperboxes of different classes. In contrast with FMCN, DCFMN needs only one type of compensatory neurons, known as overlapping neurons (OLNs), to handle both overlapped and containment regions to classify data patterns. Two different fuzzy membership functions for OLN and CLNs are also presented based on the geometric center and data core of the hyperbox.

In 2012, Devtalab et al. [18] proposed a modified version of FMCN [66], called a Modified Fuzzy Min-Max Classifier Using Compensatory Neurons (MFMCN), to handle overlapping regions problems. FMCN adds compensatory neurons straight after occurring of overlap region between hyperboxes due to the expansion step. But, MFMCN first creates all hyperboxes and after that adds compensatory neurons based on the overlap region between hyperboxes. This way results in a decrease in time and space complexity against FMCN.

In 2014, Devtalab et al. [17] proposed a novel FMNN, known as Multi-Level Fuzzy Min-Max Neural Network (MLF) classifier employing a multi-level tree structure to classify the pattern. Each node in MLF is known as a subnet and works as an independent classifier to classify patterns belonging to the particular region (overlap region in a node). At the first level (root node), the classifier is reliable to distinguish the non-boundary region (non-overlap region) patterns. Classifier at the second level is responsible for the remaining regions (overlapped region) of the root subnet (belongs to the first level). Similarly, each node in the level (except the first level) is responsible for classifying patterns belonging to an overlapped region in the previous level (parent node) in the network. Consequently, each level of the model operates in various sizes of hyperboxes to handle the overlap region.

In 2015, Forghani et al. [26] proposed an extension of FMNN, called FMNN for Learning a Classifier with Symmetric Margin (FMNWSM). FMNWSM avoids using the contraction process and additional compensatory nodes to deal with overlapped regions. The authors proposed a fuzzy membership function based on the radius and midpoint of the hyperbox. FMNWSM performs better in classification accuracy when the training and testing data are from an identical probability distribution; however, it is not practically possible to use large real-world data.

The above-mentioned approaches do not use the contraction step and provide additional structures in FMNN for decision-making in overlapped regions, overcoming the contraction's problem with the cost of an increase in training complexity of FMNN. Although these approaches (without contraction) are an enhanced version of the original FMNN to reduce classification error, these FMNN variants increase the cardinality of hyperboxes, which inturn increases the time and space complexity of training and testing phases.

3.2 Motivation

Although these FMNN variants include various improvements and enhancements on original FMNN to increase classification performance, they still exhibit certain limitations that affect FMNN classification performance negatively. These limitations are summarized in two aspects.

First, the improved and enhanced version of FMNN that include contraction procedures in learning process such as GFMNN [28], WFMNN [26], EFMNN [63], KnFMNN [64], EFMNN-2 [65] and IFMNN [52], are introduced to enhance the classification performance. However, these variants still employ the same contraction process of FMNN with few improved versions that tempered the acquired knowledge in the boundary region (overlapped and non-overlapped region) and caused gradation error in pattern classification. Even it creates classification errors for the learned data itself.

Second, the information loss due to the contraction procedure of FMNN leads to several improvements in literature to work without contraction procedures during learning such as EFC [4], EEFC [89], FMCN [66], MFMCN [18], DCFMN [131], FMNWSM [26] and MLF [17]. These approaches do not use the contraction steps. Still, they have embedded more complex structures within the simple architecture of FMNN for handling the overlapped regions that increase the cost of training and the cardinality of hyperboxes.

K-Nearest Neighbors algorithm (kNN) [25] is a supervised and non-parametric classification learning technique in the field of pattern recognition, data mining and machine learning. kNN classification algorithm doesn't have any training phase but performs an expensive testing phase for each test pattern. In kNN, each test pattern must compute the euclidean distance measure with all the training patterns. The nearest k training patterns are selected as the nearest neighbours. Based on the classes of those k nearest neighbours, voting is conducted, and the test pattern is characterized by the majority class of nearest neighbours patterns. However, in the presence of large training data, kNN requires significant testing time, making the procedure significantly expensive.

This motivated us to explore the methodology that combines the simple structure of FMNN and kNN strategy for inducing a better classification model without resorting to modifying the structure of FMNN and not including the contraction procedure. The combined hybrid model overcomes the limitation of individual models and minimizes the complexities of each of the individual models.

We have chosen kNN intentionally for decision-making in the overlapping region instead of other non-linear classification models like SVM[14], Backpropagation Neural Network [33]. As kNN is employed in the testing phase, it retains all the advantages of FMNN training, especially FMNN suitability for online adaptation.

3.3 Proposed kNN-FMNN Algorithm

Traditional FMNN with contraction steps [102], described in Section 2.2, results in non-overlapping among the hyperboxes of different classes. However, there is an information loss in the contracted boundary region; even there is a possibility that objects of one class are being absolute members of hyperboxes of another class. The defuzzification of the overlapping region affects the generalizability of the FMNN. The existing approaches dealing with the representation of overlapping regions by avoiding overlapping and contraction procedures increase the complexity of FMNN structure.

This work presents a hybridization of FMNN with kNN algorithm for handling decision-making in the overlapped region without altering the structure of FMNN. In one way, we are solving the problem of FMNN, and in another way, with the advantage of FMNN learning and results in granulation of training data through hyperboxes, we minimize the complexity of kNN classification algorithm.

Here, we introduce three changes to the traditional FMNN for enhancement. The first two modifications are in the training phase, and the third part is in the testing phase. In the training phase, first, we eliminate the contraction procedure to protect the dimensions of overlapped hyperboxes. Second, we have relaxed the k-nearest hyperbox expansion rule in papers [52, 64] to the maximum possibility. This way, we avoid the creation of too many hyperboxes that reduce the network complexity. In traditional FMNN, if the winning hyperbox with the highest membership value, out of hyperboxes corresponding to the same decision class, does not meet the expansion criterion to include the input pattern, then a new point hyperbox is created. Here, we provide the opportunity to the vicinity of winning hyperbox, which means hyperbox with the next highest membership value is checked for expansion. This process continues until any existing hyperbox can include the input pattern. If all hyperbox of same class are failed to expand, then a new point hyperbox is created. Here, we give a maximum chance for existing hyperboxes to expand fully to avoid creating new hyperboxes. These modifications also help in reducing the sensitivity of FMNN to order in which the training data is presented.

FMNN gives a natural way to group the nearest objects into the granular structure of a hyperbox. So, in this chapter, we restrict the space within hyperboxes in which kNN computation needs to be performed to classify test patterns. Here, we are utilizing the vicinity of the overlapping region in FMNN testing phase described below.

The rest of the section described the training and testing phases of kNN-FMNN algorithms given in Algorithm [3.1] and Algorithm [3.2] respectively.

3.3.1 Training of kNN-FMNN Algorithm

Let $DT = (U, C^n \cup \{d\})$ be decision system where U represents a set of training patterns, C^n is a set of numeric conditional attributes and $\{d\}$ represents a single decision attribute.

Table 3.1: Description of Function Name and Notation in Algorithm [3.1] and Algorithm [3.2]

Notation	Meaning
FM	Represents FMNN learning model.
$Memb_H(x)$	Returns membership value of x on H using Eqn. (2.2).
$FM.Belongs(x)$	Checks absolute memberhsip value of x on any existing hyperboxes of same class and return a hyperbox.
$FM.ObjSave(H, x)$	Includes object x in hyperbox H.
$FM.Save(HBS, H)$	Save H in HBS set.
$FM.HMemb(x)$	Returns a set of hyperboxes with their membership value correspond to x of same class label.
$FM.Exp(H, x)$	Checks expansion of H to include x using expansion criterion Eqn. (2.6).
$FM.Update(HBS, H)$	Updates the expanded hyperbox H in the set HBS.
$FM.Expand(H, x)$	Expands the hyperbox H to include x using Eqn. (2.7) and Eqn. (2.8).
$Break$	Breaks current loop.
$FM.Create(x)$	Creates a new point hyperbox to include x.
$AbsMemb(x, HBS)$	Returns a set of hyperboxes which have full membership for the object x using Eqn. (2.2).
$pure(HB)$	Checks whether all hyperboxes in HB that contain x correspond to the same decision class or not.
$ObjMemb(H)$	Returns the objects belonging to particular hyperbox H.
$LocalSet(HB)$	Collecting all objects belonging to all hyperboxes in HB.
$knnLocal(HO, x)$	Computing kNN on objects belonging HO for testing object x.

Let HBS is a set of hyperboxes and FM represents FMNN learning model. Initially, HBS is an empty set, and as training proceeds, hyperboxes are added to FM model, as described in Section 2.2. For each hyperbox H, the stored information is min point and max point along with objects indices having full membership into H. We preserve the object indices in kNN-FMNN for the purpose of the testing phase. Only the expansion step is performed for each input pattern x belonging to DT to preserve the overlapping region. Based on FMNN expansion criteria given in Eqn. (2.6), hyperbox can expand non-uniformly in a different dimension as cumulative widths of all dimensions need to be less than $n\theta$.

Based on Algorithm [3.1], for every training pattern x, $Belongs(x)$ finds fuzzy membership value of x with all hyperboxes representing the same class using Eqn. (2.2) and return a hyperbox (H) with full membership value of one to x.

$$Belong(x) = \{H \mid H \in HBS \wedge Memb_H(x) == 1 \wedge d(x) == d(H)\} \tag{3.1}$$

If $Belongs(x)$ is non-empty, then x is added to the particular hyperbox (H) giving the full

Algorithm 3.1: Training of kNN-FMNN

```
Input  : DT: Training Samples, γ: Gamma parameter, θ: Theta parameter
Output: HBS: Collection of hyperboxes of different classes, Learning model FM.
Let HBS = ∅;
for every x in DT do
    if FM.Belong(x) ≠ ∅ then
        H = FM.Belong(x);
        FM.ObjSave(H,x);
    else
        HS = FM.HMemb(x);
        Flag = 0;
        if HS ≠ ∅ then
            for every H in HS do
                if Exp(H, x) == True then
                    FM.Expand(H,x);
                    FM.Update(HBS,H);
                    Flag = 1;
                    Break;
                end
            end
            if Flag == 0 then
                H=FM.Create(x);
                FM.ObjSave(H,x);
                FM.Save(HBS,H);
            end
        else
            H=FM.Create(x);
            FM.ObjSave(H,x);
            FM.Save(HBS,H);
        end
    end
end
return HBS, FM
```

membership without modifying the hyperbox using $ObjSave(H, x)$. Otherwise, $HMemb(x)$ gives a list of hyperboxes HS with membership value. For each $H \in HS$ with the highest membership value, if $Exp(H, x)$ (Expansion Criteria) is satisfied, the hyperbox H is expanded using Eqn. (2.7) and Eqn. (2.8) and object x saved to the hyperbox H using $ObjSave(H, x)$. If $Exp(H, x)$ is not satisfied on a particular hyperbox H, then the next hyperbox in HS with the highest membership hyperbox is checked for expansion whether it includes input pattern x or not. This process continues until any hyperbox that can include the input pattern. If none of the hyperboxes in HS are met expansion criteria or $HMemb(x)$ returns an empty set, a point hyperbox H is created using $Create(x)$, and x is added to the point hyperbox created using $ObjSave(H, x)$ and resulting H is added in HBS.

3.3.2 Testing of kNN-FMNN Algorithm

Let DS be a set of testing samples. Based on Algorithm [3.2], for every testing pattern x in DS, we compute the fuzzy membership value w.r.t. all hyperboxes HBS in FM model. Because the overlapping among hyperboxes is allowed in the training phase due to eliminating contraction step and extended expansion criteria, it is possible to obtain absolute membership of one to multiple hyperboxes. $AbsMemb(x)$ returns all the hyperboxes (HB) giving full membership value.

$$AbsMemb(x, HBS) = \{H \mid H \in HBS \wedge Memb_H(x) == 1\} \tag{3.2}$$

If HB set is empty, then the testing pattern x does not belong to any of the hyperboxes and a decision is taken like traditional FMNN testing by assigning the decision class corresponding to the nearest hyperbox; otherwise, the purity of the collection is examined using $pure(HB)$.

$$pure(HB) = \left| \bigcup_{H \in HB} \{class(H)\} \right| == 1 \tag{3.3}$$

The resulting collection is pure only if all hyperboxes correspond to a single decision class, in which case, without ambiguity, that class is assigned to the testing pattern. In the case of impurity, objects belonging to all these hyperboxes using $ObjMemb(H)$ are collected in $LocalSet(HB)$ set and then applied kNN using $knnLocal$ function on these objects locally to determine the decision class of x.

$$LocalSet(HB) = \bigcup_{H \in HB} (ObjMemb(H)) \tag{3.4}$$

3.4 Complexity Analysis of kNN-FMNN Algorithm

This section shows the time and space complexity analysis of the proposed algorithm kNN-FMNN. The following variables are used in the complexity analysis of kNN-FMNN.

- $|U|$: the number of objects.
- $|HBS|$: the number of hyperboxes.
- $|C^n|$: the number of numeric conditional attribute.

Algorithm 3.2: Testing of kNN-FMNN

```
   Input  : DS: Testing Samples, Learning Model FM, HBS: Set of hyperboxes, k:
            k-nearest neighbour value
 1 for every x in DS do
      // Compute fuzzy membership value of x w.r.t.  all hyperboxes in HBS
 2    HB = FM.AbsMemb(x, HBS);
 3    if HB ≠ ∅ then
 4       if pure(HB) == True then
 5          Classify x as decision class of HB;
 6       else
 7          HObj = FM.LocalSet(HB);
 8          Classify x as decision class of FM.knnLocal(HObj, x);
 9       end
10    else
11       Classify x to highest membership hyperbox decision class;
12    end
13 end
```

Table 3.2 shows the time complexity of the proposed algorithm. In the Table, Algorithm 3.1 (training phase) with steps 1-29 computes for each training data to check complete belonging to the existing hyperboxes or modifying of hyperbox or creation of hyperbox with time complexity $O(|U| * |HBS| * |C^n|)$. In Table, from steps 1-12 in Algorithm 3.2 (Testing phase) performs to classify a test pattern by checking the belongingness over trained hyperboxes $|HBS|$ with time complexity $O(|HBS| * |C^n|)$. If selected hyperboxes where a test pattern exist are in different classes, steps 7-8 are computed to classify a test pattern using kNN with complexity $O(|U| * |C^n|)$. Hence, the time complexity of the algorithm (testing phase) for each test pattern is obtained: $O(|HBS| * |C^n|) + O(|U| * |C^n|) = O(|U| \times |C^n|)$ since, $|HBS| << |U|$.

So, the total complexity of the proposed algorithm kNN-FMNN, including both the training and testing phase, is: $O(|U| * |HBS| * |C^n|) + O(|U| * |C^n|)$.

The entire decision system must be loaded first in memory for computation of kNN-FMNN algorithm. For preserving FMNN model, $O(|HBS| * |C^n|)$ space is needed, and for kNN process where data needs to be available, $O(|U| * |C^n|)$ space is required. Thus, the space complexity of kNN-FMNN algorithm is $O(|HBS| * |C^n|) + O(|U| * |C^n|) = O(|U| \times |C^n|)$ since, $|HBS| << |U|$.

Table 3.2: Time Complexity Analysis of kNN-FMNN

Algorithm (phase)	**Steps in Algorithm**	**Time complexity**
Training (Algorithm 3.1)	2-29. Creation of hyperboxes	$O(\lvert U\rvert * \lvert HBS\rvert * \lvert C^n\rvert)$
Testing (Algorithm 3.2)	1-13. Testing on hyperboxes	$O(\lvert HBS\rvert * \lvert C^n\rvert)$
	7-8. Testing on overlap region with kNN	$O(\lvert U\rvert * \lvert C^n\rvert)$

Table 3.3: Benchmark Datasets

Dataset	**Attributes**	**Objects**	**Classes**
Ionosphere	32	351	2
Vehicle	18	846	4
Segment	16	2310	2
Steel	27	1941	7
Ozone Layer	72	1848	2
Page	10	5472	5
Robot	24	5456	4
Waveform2	40	5000	3
Texture	40	5500	11
Gamma	10	19020	2
Satimage	36	6435	6
Ring	20	7400	2
Musk2	166	6598	2
Shuttle	9	57999	7
Sensorless	48	58509	11
MiniBooNE	50	129596	2
Winnipeg	174	325834	7

3.5 Experiments and Results

The system configuration used for experimentation are CPU: Intel(R) i7-8500, Clock Speed: 3.40GHz × 6, RAM: 32 GB DDR4, OS: Ubuntu 18.04 LTS 64 bit and Software: Matlab R2017a. The detailed experimental evaluation is conducted on seventeen benchmark numeric decision systems taken from UCI machine learning repository [21], the details are given in Table 3.3. The proposed algorithm kNN-FMNN is implemented in the Matlab environment. In our experiments, we set the sensitive parameter γ value equal to 4, as recommended in the paper [63, 102].

We have experimented kNN-FMNN with different theta (θ) values, and all the results were not reported due to space constraints. It is observed that small theta values such as 0.01 or 0.02 create the large cardinality of hyperboxes that may avoid data overfitting but with circumstances where each object or input is learned as individual hyperboxes. On the other hand, the large theta values such as 0.85 or 0.9 create fewer cardinality of hyperboxes but gradually decrease the capability to capture nonlinear separability boundaries between multiple classes, and also is almost like a kNN algorithm because of the large boundary region. So, our objective is to select theta values that minimize the cardinality of hyperboxes and optimize the classification performance. Hence, on results obtained from conducted experiment empirically, we have finalized the value of theta as 0.3. This value is adopted throughout the thesis. And, k in kNN is set to be 3, which was found sufficient for the testing phase. The performance of kNN-FMNN is examined through a comparative evaluation through ten-fold cross-validation (10-FCV).

3.5.1 Relevance of Proposed Approach through 10-FCV

This section assesses the performance of the proposed algorithm kNN-FMNN by comparing it with the original FMNN and some popular variants of FMNN approaches, such as GFMNN [28], EFMNN [63], MLF [17] and IFMNN [52]. We implement the comparative algorithms i.e., GFMNN [28], EFMNN [63], and IFMNN [52] in the Matlab environment, and MLF code is provided by author [17] in Matlab. We set the sensitive parameter γ and theta value to 4 and 0.3, respectively. The comparative experiments are conducted in the same system using the Matlab environment.

10-FCV based comparative experiment is conducted to assess the performance of kNN-FMNN. 10-FCV is performed on the original dataset to comprehend the model's ability. In 10-FCV, the original dataset is partitioned into ten subsets. In each iteration, one subset is retained for the testing part, and the remaining nine subsets are used for training the model.

Furthermore, a paired t-test with a significance level of 0.05 is performed to analyze the statistical evaluation of kNN-FMNN results over given compared algorithms. Each column in Tables 3.4, 3.5, 3.6 reports the results of the respective algorithm in the form of mean and standard deviation along with *p-value* except kNN-FMNN column. kNN-FMNN column contained only mean and standard deviation. The *p-value* index is the significant level between the respective algorithm with kNN-FMNN algorithm. For classification, if *p-value* is greater than 0.05, then there is no statistically significant difference, marked with the symbol 'o'. If *p-value* is less than 0.05, and the result obtained by the respective algorithm is less than kNN-FMNN, then the particular algorithm is statistically inferior to kNN-FMNN and marked as a loss '-'. Otherwise, it represents a win '+'. The contrary measure is for computational time and obtained hyperboxes which means if the *p-value* is less than 0.05, and the result obtained by the respective algorithm is less than kNN-FMNN, then the particular algorithm is statistically significant than kNN-FMNN and marked as a win '+'; otherwise, it represents a loss '-'.

The last three lines in each Table 3.4, 3.5 and 3.6 correspond to **Average (NOD)**, **CAverage**, and **Lose/Win/Tie**. It can be observed that the datasets over which an algorithm is executing vary from one to another. Hence, the average of individual mean values is reported in two forms. Average (NOD) corresponds to the average value obtained by an algorithm on datasets where it could be evaluated along with reporting the number of datasets (NOD) involved in brackets. CAverage value depicts the average of the individual mean obtained by restricting to only those datasets in which all algorithms could be evaluated. For the comparative analysis, CAverage plays an important role. The last line indicates the count of the number of statistically loss('-'), better('+'), and equivalent('o') for each algorithm in comparison with the proposed kNN-FMNN.

Table 3.4 reports the comparative classification accuracy results of classifiers based on 10-FCV. Also, Table 3.5 and Table 3.6 show the computational time results and number of obtained hyperboxes results by respective algorithms on 10-FCV. Fig. 3.2, Fig. 3.3 and Fig. 3.4 depict the box-plot representation of results given in Table 3.4, Table 3.5 and Table 3.6 respectively.

Note: In the comparative experiment, the computational results include both training and testing phase times. '#' sign in each Tables 3.4, 3.5 and 3.6 represents the scenario of non-termination of the code even after several hours of computation. In all Figures 3.2, 3.3 and 3.4, the range of Y-axis varies based on obtained results in each dataset.

3.5.2 Analysis of Results

Classification Results

Table 3.4 and Fig. 3.2 present the obtained classification accuracy results in 10-FCV. Based on Table 3.4, kNN-FMNN reached the highest CAverage value (91.13) than compared approaches. kNN-FMNN obtained statistically significant or similar results than compared algorithms in most of the datasets. Considering the overall 75 accuracy results across all the compared algorithms, the cumulative lose/win/tie results as 56/3/16. Hence, in the majority of 56 results, the proposed algorithm kNN-FMNN performed significantly better than the compared algorithms. Only in 3 results (in Steel dataset by EFMNN, in Robot dataset by GFMNN, in Gamma by MLF), the compared algorithms performed statistically better than kNN-FMNN. And in the remaining 16 results, kNN-FMNN performed statistically similar to compared algorithms.

This significantly validates the utility of hybridization of kNN and FMNN through the proposed kNN-FMNN algorithm in obtaining better generalization.

Table 3.4: Classification Accuracies Results with Classifiers in 10-FCV

Datasets	**kNN-FMNN**	**FMNN**		**GFMNN**		**EFMNN**		**MLF**		**IFMNN**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	92.87 ± 3.63	91.17 ± 6.65	0.49^{o}	90.90 ± 4.96	0.32^{o}	88.62 ± 4.19	0.03^{-}	88.04 ± 4.61	0.02^{-}	84.04 ± 6.36	0.00^{-}
Vehicle	70.54 ± 6.06	62.54 ± 3.96	0.00^{-}	66.28 ± 3.32	0.07^{o}	61.87 ± 7.30	0.01^{-}	70.56 ± 5.54	1.00^{o}	61.07 ± 4.01	0.00^{-}
Segment	96.45 ± 1.70	89.78 ± 1.77	0.00^{-}	89.26 ± 2.03	0.00^{-}	95.45 ± 1.14	0.14^{o}	95.89 ± 1.72	0.47^{o}	86.19 ± 3.85	0.00^{-}
Steel	91.45 ± 2.05	91.55 ± 1.46	0.90^{o}	88.82 ± 2.40	0.02^{-}	93.30 ± 1.75	0.04^{+}	93.04 ± 1.84	0.08^{o}	84.75 ± 4.04	0.00^{-}
Ozone	95.71 ± 1.72	82.57 ± 4.66	0.00^{-}	90.37 ± 2.64	0.00^{-}	93.76 ± 2.29	0.05^{-}	96.64 ± 1.33	0.19^{o}	90.19 ± 3.60	0.00^{-}
Page	96.25 ± 1.00	89.20 ± 1.76	0.00^{-}	92.01 ± 1.73	0.00^{-}	92.18 ± 0.98	0.00^{-}	95.50 ± 0.93	0.10^{o}	66.26 ± 21.46	0.00^{-}
Robot	93.00 ± 0.84	91.68 ± 1.35	0.02^{-}	94.19 ± 0.84	0.01^{+}	93.79 ± 1.07	0.08^{o}	93.55 ± 0.86	0.17^{o}	82.10 ± 2.21	0.00^{-}
Waveform2	79.36 ± 1.21	67.48 ± 1.93	0.00^{-}	69.76 ± 2.32	0.00^{-}	63.02 ± 3.45	0.00^{-}	74.34 ± 2.55	0.00^{-}	68.06 ± 3.11	0.00^{-}
Texture	95.36 ± 0.61	80.65 ± 1.69	0.00^{-}	87.38 ± 2.06	0.00^{-}	85.49 ± 1.43	0.00^{-}	95.00 ± 1.20	0.40^{o}	86.29 ± 1.97	0.00^{-}
Ring	91.93 ± 0.35	56.81 ± 1.88	0.00^{-}	79.85 ± 2.10	0.00^{-}	58.77 ± 3.10	0.00^{-}	79.45 ± 2.12	0.00^{-}	70.34 ± 3.03	0.00^{-}
Gamma	81.11 ± 0.85	67.41 ± 2.80	0.00^{-}	74.36 ± 2.05	0.00^{-}	72.66 ± 3.32	0.00^{-}	82.06 ± 0.92	0.03^{+}	70.49 ± 2.67	0.00^{-}
Satimage	88.19 ± 1.40	80.39 ± 1.09	0.00^{-}	82.98 ± 1.36	0.00^{-}	81.94 ± 1.08	0.00^{-}	87.77 ± 1.54	0.53^{o}	81.24 ± 2.50	0.00^{-}
Musk2	96.47 ± 0.85	90.91 ± 2.41	0.00^{-}	92.30 ± 1.06	0.00^{-}	92.20 ± 1.09	0.00^{-}	92.97 ± 1.09	0.00^{-}	90.42 ± 1.41	0.00^{-}
Shuttle	99.92 ± 0.05	81.83 ± 11.14	0.00^{-}	94.96 ± 3.44	0.00^{-}	68.80 ± 13.16	0.00^{-}	99.94 ± 0.03	0.21^{o}	96.24 ± 8.37	0.18^{o}
Sensorless	98.44 ± 0.15	61.01 ± 3.29	0.00^{-}	59.10 ± 2.41	0.00^{-}	80.54 ± 1.67	0.00^{-}	97.05 ± 0.23	0.00^{-}	59.47 ± 1.22	0.00^{-}
MiniBooNE	89.59 ± 0.32	#		#		#		#		#	
Winnipeg	98.08 ± 0.09	#		#		#		#		#	
Average (NOD)	91.49 (17)	80.14 (15)		84.44 (15)		82.36 (15)		89.14 (15)		79.60 (15)	
***CAverage*$^{\$}$**	91.13	80.14		84.44		82.36		89.14		79.60	
Lose/Win/Tie		13/0/2		12/1/2		12/1/2		5/1/9		14/0/1	

NOD: Number of datasets over which the average is computed (indicated in bracket).

$: Average mean value over 15 datasets where all algorithms executed.

represents non-termination of program even after several hours.

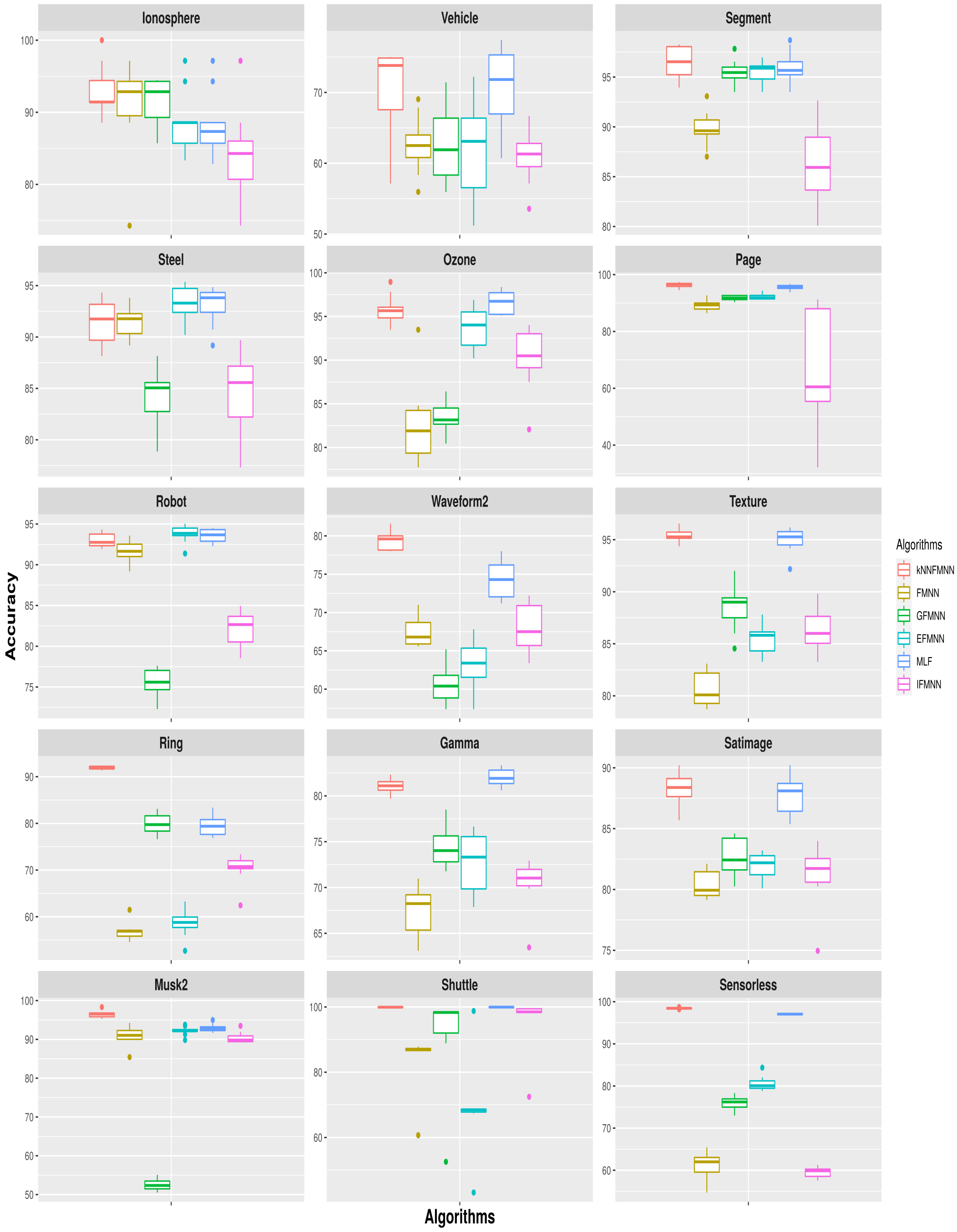


Figure 3.2: Boxplots for Classification Accuracies Results of Table 3.4

Table 3.5: Computational Time (in Seconds) Results in 10-FCV

Datasets	**kNN-FMNN**	**FMNN**		**GFMNN**		**EFMNN**		**MLF**		**IFMNN**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	0.03 ± 0.00	0.06 ± 0.01	0.00^-	0.05 ± 0.01	0.00^-	0.08 ± 0.00	0.00^-	0.13 ± 0.01	0.00^-	0.08 ± 0.01	0.00^-
Vehicle	0.03 ± 0.01	0.20 ± 0.02	0.00^-	0.14 ± 0.02	0.00^-	0.28 ± 0.01	0.00^-	5.94 ± 0.24	0.00^-	0.24 ± 0.03	0.00^-
Segment	0.03 ± 0.01	0.45 ± 0.03	0.00^-	0.22 ± 0.03	0.00^-	1.08 ± 0.03	0.00^-	10.83 ± 0.73	0.00^-	0.36 ± 0.03	0.00^-
Steel	0.46 ± 0.03	2.92 ± 0.32	0.00^-	1.46 ± 0.11	0.00^-	2.56 ± 0.16	0.00^-	27.94 ± 2.13	0.00^-	2.30 ± 0.24	0.00^-
Ozone	1.21 ± 0.03	4.74 ± 0.07	0.00^-	4.39 ± 0.06	0.00^-	4.63 ± 0.03	0.00^-	18.54 ± 1.13	0.00^-	4.83 ± 0.07	0.00^-
Page	0.05 ± 0.01	0.58 ± 0.13	0.00^-	0.13 ± 0.01	0.00^-	2.54 ± 0.83	0.00^-	11.00 ± 0.48	0.00^-	0.78 ± 0.19	0.00^-
Robot	1.86 ± 0.03	9.30 ± 0.33	0.00^-	4.38 ± 0.15	0.00^-	15.95 ± 0.10	0.00^-	1094.80 ± 28.15	0.00^-	12.50 ± 0.77	0.00^-
Waveform2	4.10 ± 0.06	11.98 ± 0.13	0.00^-	11.82 ± 0.18	0.00^-	11.98 ± 0.07	0.00^-	5156.95 ± 91.81	0.00^-	11.97 ± 0.18	0.00^-
Texture	0.05 ± 0.01	1.37 ± 0.04	0.00^-	0.64 ± 0.04	0.00^-	1.71 ± 0.16	0.00^-	8.07 ± 0.42	0.00^-	0.96 ± 0.14	0.00^-
Ring	4.64 ± 0.06	21.96 ± 0.24	0.00^-	909.30 ± 24.14	0.00^-	22.01 ± 0.48	0.00^-	253.64 ± 8.68	0.00^-	18.29 ± 0.43	0.00^-
Gamma	1.25 ±0.02	65.36 ±6.80	0.00^-	839.40 ±85.27	0.00^-	76.62 ±3.81	0.00^-	586.44 ±36.72	0.00^-	118.04 ± 8.66	0.00^-
Satimage	0.24 ± 0.02	5.53 ± 0.25	0.00^-	2.38 ± 0.16	0.00^-	5.34 ± 0.48	0.00^-	163.80 ± 10.35	0.00^-	5.81 ± 0.83	0.00^-
Musk2	10.38 ± 0.75	53.88 ± 3.50	0.00^-	35.47 ± 2.25	0.00^-	57.70 ± 2.20	0.00^-	2275.16 ± 137.54	0.00^-	84.44 ± 4.61	0.00^-
Shuttle	0.30 ± 0.01	3.22 ± 0.47	0.00^-	1.54 ± 0.09	0.00^-	8.24 ± 1.34	0.00^-	14.78 ± 1.19	0.00^-	555.12 ± 212.34	0.00^-
Sensorless	0.34 ± 0.01	56.27 ± 23.71	0.00^-	7.36 ± 3.45	0.00^-	209.47 ± 13.75	0.00^-	2094.55 ± 38.88	0.00^-	7.20 ± 0.65	0.00^-
MiniBooNE	91.72 ± 0.66	#		#		#		#		#	
Winnipeg	185.67 ± 1.75	#		#		#		#		#	
Average (NOD)	17.78 (17)	216.50 (15)		113.68 (15)		26.38 (15)		736.20 (15)		51.46 (15)	
CAverage$^{\$}$	1.66	216.50		113.68		26.38		736.20		51.46	
Lose/Win/Tie		15/0/0		15/0/0		15/0/0		15/0/0		15/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).

\$: Average mean value over 15 datasets where all algorithms executed.

represents non-termination of program even after several hours.

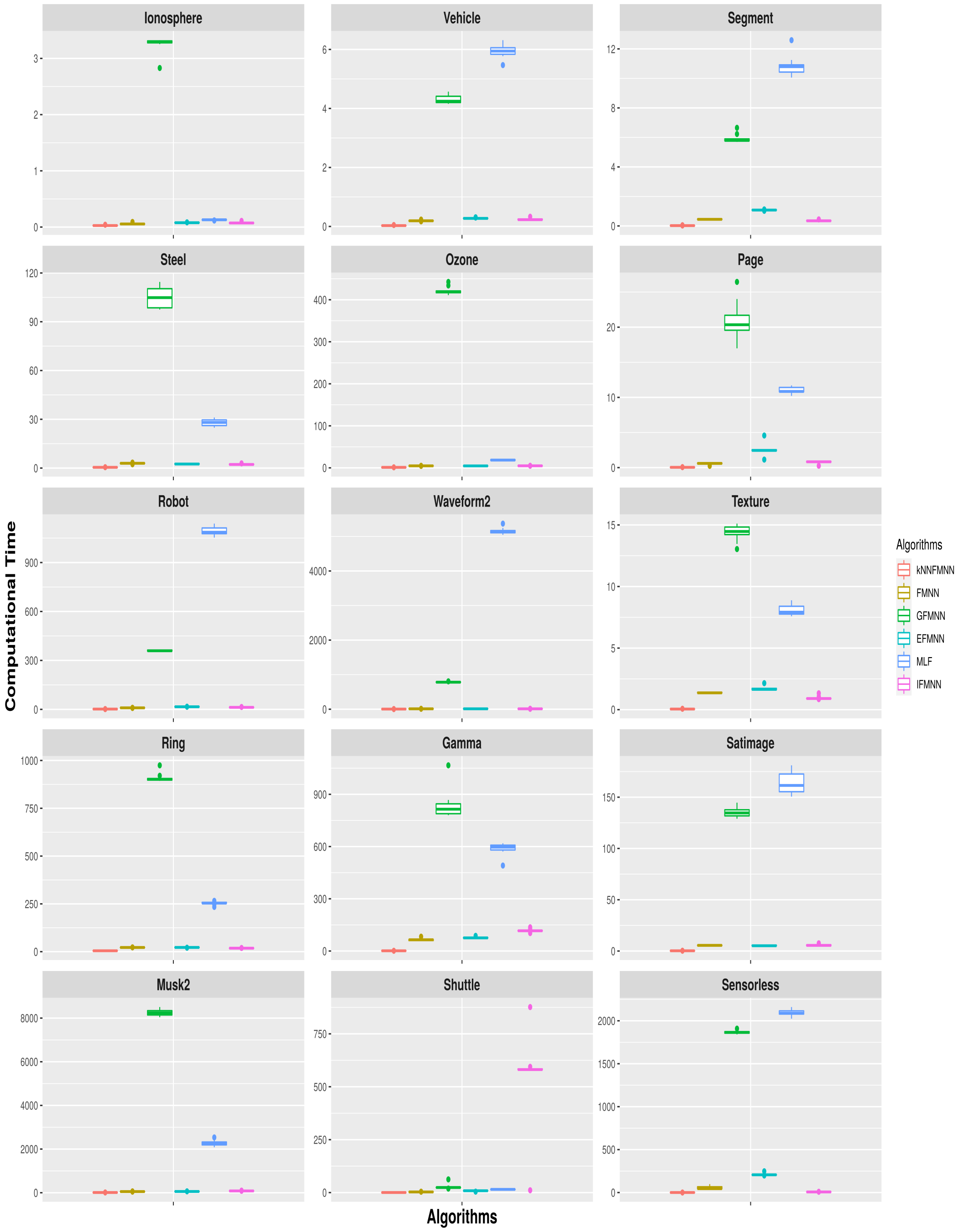


Figure 3.3: Boxplots for Computational Time Results of Table 3.5

Table 3.6: Obtained Cardinality of Hyperboxes in 10-FCV

Datasets	**kNN-FMNN**	**FMNN**		**GFMNN**		**EFMNN**		**MLF**		**IFMNN**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	80.00 ± 1.94	173.60 ± 9.80	0.00^{-}	126.80 ± 9.96	0.00^{-}	219.50 ± 9.11	0.00^{-}	157.10 ± 2.47	0.00^{-}	230.80 ± 3.19	0.00^{-}
Vehicle	70.60 ± 3.31	487.50 ± 12.86	0.00^{-}	245.20 ± 6.01	0.00^{-}	523.90 ± 5.97	0.00^{-}	209.70 ± 9.25	0.00^{-}	418.90 ± 16.18	0.00^{-}
Segment	33.00 ± 0.82	406.10 ± 11.51	0.00^{-}	108.60 ± 10.77	0.00^{-}	700.20 ± 16.00	0.00^{-}	177.90 ± 4.28	0.00^{-}	139.30 ± 5.29	0.00^{-}
Steel	183.70 ± 3.09	1324.00 ± 69.62	0.00^{-}	786.10 ± 19.66	0.00^{-}	1068.10 ± 12.10	0.00^{-}	598.80 ± 3.74	0.00^{-}	1148.00 ± 39.18	0.00^{-}
Ozone	287.90 ± 3.78	1571.80 ± 17.26	0.00^{-}	1441.50 ± 13.82	0.00^{-}	1564.10 ± 8.06	0.00^{-}	673.10 ± 5.32	0.00^{-}	1520.80 ± 11.56	0.00^{-}
Page	26.00 ± 1.41	473.30 ± 92.60	0.00^{-}	77.20 ± 11.36	0.00^{-}	726.10 ± 66.81	0.00^{-}	663.10 ± 55.56	0.00^{-}	62.00 ± 15.49	0.00^{-}
Robot	552.50 ± 7.75	2054.10 ± 14.00	0.00^{-}	812.30 ± 28.19	0.00^{-}	2725.90 ± 11.62	0.00^{-}	2108.80 ± 5.45	0.00^{-}	3851.30 ± 120.66	0.00^{-}
Waveform2	958.60 ± 4.79	4489.70 ± 0.95	0.00^{-}	4468.90 ± 3.48	0.00^{-}	4482.60 ± 2.22	0.00^{-}	2509.80 ± 9.91	0.00^{-}	4449.40 ± 4.30	0.00^{-}
Texture	47.50 ± 1.27	1313.80 ± 48.21	0.00^{-}	228.80 ± 20.13	0.00^{-}	680.70 ± 13.12	0.00^{-}	970.20 ± 43.28	0.00^{-}	353.90 ± 34.57	0.00^{-}
Ring	627.00 ± 3.80	5327.00± 45.21	0.00^{-}	4535.70± 100.78	0.00^{-}	5235.80 ± 52.47	0.00^{-}	6400.00 ± 276.85	0.00^{-}	4081.80 ± 34.81	0.00^{-}
Gamma	309.50 ± 4.79	9179.90 ± 277.30	0.00^{-}	3441.20 ± 248.78	0.00^{-}	6996.70 ± 343.20	0.00^{-}	17401.80 ± 1004.00	0.00^{-}	6897.70 ± 464.42	0.00^{-}
Satimage	216.60 ± 7.15	3310.50 ± 43.23	0.00^{-}	1675.60 ± 42.69	0.00^{-}	2638.1 ± 43.68	0.00^{-}	3320.8 ± 113.04	0.00^{-}	2327.4 ± 79.02	0.00^{-}
Musk2	739.30 ± 11.03	4072.10 ± 95.31	0.00^{-}	2791.20 ± 43.02	0.00^{-}	4337.2 ± 7.08	0.00^{-}	3667.4 ± 9.02	0.00^{-}	5828.9 ± 7.02	0.00^{-}
Shuttle	14.40 ± 0.84	46.00 ± 4.74	0.00^{-}	15.40 ± 0.84	0.02^{-}	87.10 ± 3.48	0.00^{-}	80.10 ± 5.26	0.00^{-}	12160.1 ± 4078.5	0.00^{-}
Sensorless	27.30 ± 1.34	2303.10 ± 887.81	0.00^{-}	142.00 ± 144.60	0.02^{-}	4458.7 ± 303.9	0.00^{-}	9432.40 ± 364.26	0.00^{-}	57.20 ± 9.09	0.00^{-}
MiniBooNE	1921.2 ± 0.01	#		#		#		#		#	
Winnipeg	3515.8 ± 0.02	#		#		#		#		#	
Average (NOD)	565.34 (17)	2313.12 (15)		1315.18 (15)		2369.31 (15)		3047.07 (15)		2747.60 (15)	
CAverage$^{\$}$	278.22	2313.12		1315.18		2369.31		3047.07		2747.60	
Lose/Win/Tie		15/0/0		15/0/0		15/0/0		15/0/0		15/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 15 datasets where all algorithms executed.
represents non-termination of program even after several hours.

Figure 3.4: Boxplots for Obtained Cardinality of Hyperboxes of Table 3.6

Computational Time Results

The computational complexity of FMNN training algorithm is proportional to the cardinality of hyperboxes created. In addition to the cardinality of hyperboxes, the cost of complex structures and procedures such as contraction steps and hierarchical layers in algorithms like GFMNN, EFMNN, IFMNN and MLF increases the complexity. The computational time reported in Table 3.5 and Fig. 3.3 validates that kNN-FMNN incurred significantly less computational time than compared algorithms on all datasets. Based on a subset of datasets where all compared algorithms can execute, the proposed method kNN-FMNN obtained the lowest CAverage value (1.66 seconds), which is significantly lesser than compared algorithms with CAverage in the range of 26 to 736 seconds.

Even the resulting standard deviation of computation time presented very little variation, thus showing that the methodology is reliable compared to others approaches. These substantial reductions in computational time of kNN-FMNN are due to adapting FMNN with only the expansion step and relaxing k-nearest hyperbox expansion rule to the maximum possibility for expansion that achieves a much lesser cardinality of hyperboxes compared to other approaches. Thus, the speed-up in computation and classification model performance demonstrates the potential of the kNN-FMNN algorithm and its suitability for larger datasets.

In MiniBooNE and Winnipeg datasets, kNN-FMNN classifier could learn in significantly less computational time, whereas compared algorithms were unable to compute. Because considering the maximum possibility of expansion rule to select hyperboxes for expansion allowed hyperboxes to expand fully or give more chances to acquire full knowledge. This way excluded from creating too many hyperboxes whereas, in compared algorithms, they imposed restrictions on expansion rule which may lead to creating many hyperboxes that are not even used and result in a need for excess memory requirement.

Cardinality of Hyperboxes Results

The obtained cardinality of hyperboxes result, shown in Table 3.6 and Fig. 3.4, is significantly lesser than compared algorithms for all given datasets because of the relaxing the k-nearest hyperbox expansion step for selecting hyperboxes for expansion that include the more training patterns and create less number of hyperboxes in the training phase. kNN-FMNN achieved the lowest CAverage value (278.22) than compared algorithms having CAverage values ranging between 1315.18 to 2747.60. The substantial reduction in hyperboxes of kNN-FMNN resulted in a significant drop in computational time.

In summary, the relevance of kNN-FMNN is significantly validated as it computes a lesser number of hyperboxes and incurs less computational time while inducing classification models with similar or better classification accuracies than the model induced through compared algorithms.

3.6 Summary

Several improvements of FMNN were proposed to overcome limitations that arise due to the contraction step. These extensions added additional complexity to FMNN, thus increasing the training time and network complexity. This work proposed kNN-FMNN as a hybridization of FMNN with kNN to overcome the contraction step while preserving the simple structure (no modification) of FMNN. The proposed kNN-FMNN method considered only expansion steps and enriched them with a relaxed expansion rule to capture the potential underlying structure of data. The proposed approach resulted in building the classification model with relatively fewer cardinality of hyperboxes and achieved good classification accuracy by utilizing kNN locally for disambiguating classification decisions in the overlapping region. Comparative experimental studies of kNN-FMNN with existing state-of-the-art approaches [17, 28, 52, 63] over benchmark datasets proved the utility of the proposed kNN-FMNN approach in terms of better classification performance incurring less computational time and obtaining the least number of hyperboxes. Also, kNN-FMNN enhanced scalability to such large decision systems, where existing state-of-the-art FMNN methods failed to execute. Our proposed hybrid model kNN-FMNN successfully lessen the limitations of individual FMNN and kNN models.

Chapter 4

Hybridization of Fuzzy Min-Max Neural Networks with Fuzzy Rough Sets for Feature Subset Selection

This chapter addresses the issue related to scalability of the existing fuzzy rough sets (FRS) approaches on large decision systems. The FRS theory provides a robust framework for feature subset selection. Indeed, the theory has been performing remarkably with further extensions and modifications in recent years. Despite having these modifications and extensions, FRS approaches suffer from their ability to scale to large decision systems due to the space complexity of FRS methodology. This chapter addresses the related issue associated with FRS methodology and proposes an algorithm that can scale to large decision systems.

The rest of the chapter is designed as follows: Section 4.1 present the brief introduction. Section 4.2 present the basic notions about rough set theory and fuzzy rough set theory with its corresponding feature subset selection methods via discernibility matrix. Section 4.3 briefly introduces the literature survey of fuzzy rough sets approaches and their disadvantages. Section 4.4 presents the motivation of the proposed algorithm. Section 4.5 describes the functioning of the proposed algorithm FDM-FMFRS. Section 4.6 describes the complexity analysis of proposed algorithm FDM-FMFRS. Section 4.7 reports a series of experiments and comparative analysis of FDM-FMFRS with state-of-the-art approaches.

4.1 Introduction

Making a decision under imprecision and uncertainty is one of the most challenging topics in the field of data analysis. Data analysis aims to find or learn hidden patterns in a dataset, which is beneficial to find dependencies. Feature selection plays an essential role in analyzing the datasets where some features might be redundant/irrelevant that degrade the performance and increase the model's computational complexity [110, 123]. Thus, it is well essential to preprocess the data to eliminate irrelevant features that negatively impact the performance of learning models. Feature selection is a primary task in many disciplines, i.e., machine learning, pattern recognition etc., for both description and prediction purposes. Feature selection remains relevant in this era of deep learning [55, 61] as it has the potential to retain the original semantics of attributes and hence useful in building interpretable classification models. Whereas deep learning-based classification models [143], even though they are successful in obtaining better classification accuracies, currently fail in achieving interpretability as they are inherently based on feature extraction.

In the 1980s, Zdzisław I. Pawlak [77] introduced the concept of classical rough set theory (RST) as a mathematical tool useful for feature selection (semantic preserving dimensionality reduction) and rule induction in the information/decision systems [56, 78, 110, 123]. RST methodology gives new momentum to data mining [132] and knowledge discovery [140], and provides a unique insight into artificial intelligence and cognitive sciences both in practical and theoretical perspectives [19, 108, 110, 116]. RST, as a soft computing paradigm, has been successfully hybridized with other soft computing models like fuzzy sets and artificial neural networks [3, 57, 74, 79].

Particularly, RST applies primarily to symbolic/categorical decision systems [77, 78, 123]. However, the application of classical rough sets to numeric decision systems produces feature subsets with finer granularity. Hence, the induced rules from the selected features suffer from poor generalizability in classification. One solution is to discretize the numeric dataset beforehand and produce a new dataset with categorical values. Discretization is a method to partition continuous attribute domains into a finite number of discrete (non-overlapping) intervals and further assigning categorical labels to intervals [29, 58, 68, 70]. The discretization process exhibits the simplification of data in more concise, compact and making learning faster.

Nevertheless, any discretization process tends to cause a loss of information and result in classification error in pattern space [69]. Even obtaining an optimal way for the discretization process in a dataset is an NP-Hard problem [69]. The choice of discretizer will affect the success of the posterior learning task in classification [105]. However, the discretization method is often inadequate and causes essential information loss to hamper subsequent feature subset selection quality. Alternate approaches were developed to deal with the hybrid decision system without relying on a discretization step by generalizing the rough set theory in different scenarios [9].

Lately, Dubois et al. [23, 87] generalize the RST into a fuzzy rough sets (FRS) using fuzzification that deals with both symbolic and real-valued conditional attributes without any need for domain-specific knowledge. This generalization provides much greater flexibility in theoretical and application viewpoints and evolves extensively to reduct computation in hybrid decision systems [124]. FRS can approximate the crisp decision concepts in the fuzzy approximation space. Thus, FRS extends the notion of rough equivalence relation into an idea of fuzzy equivalence relation or a fuzzy tolerance relation, resulting in a fuzzy partition of the universe U.

The basic idea of the fuzzy rough model is to induce a fuzzy similarity relation, which can further be used in the construction of the fuzzy lower/upper approximation and construction of discernibility relation of a given decision system [32]. The sizes of the lower and upper approximation reflect the discriminating capability of a feature subset. The union of fuzzy lower approximation forms the fuzzy positive region of decision. The fuzzy dependency is defined as the ratio of the sizes of fuzzy positive regions over all samples in the feature space. It is used to evaluate the significance of a computed subset of features.

Skowron et al. [103, 144] introduced a feature selection mechanism based on the concept of crisp discernibility matrix (DM) in the context of Pawlak's RST. The idea of discernibility matrix construction establishes a theoretical and logical foundation for reduct computation on symbolic decision systems. Though finding all/minimal reducts with these techniques is an NP-Hard problem, these methods provide a crucial mathematical foundation for reduct computation [10, 122]. Even though these approaches can be guaranteed to obtain the exact reduct, they require a substantial computational complexity for large datasets. Jensen et al. [38] further extended the crisp DM into the fuzzy DM to determine the FRS reducts.

Various FRS reduct algorithms have been developed to perform reduct computation. These methods include information entropy based [34, 136, 137], dependency function-based [38, 42, 43, 112, 113] and discernibility matrix (DM) based [10, 15, 38, 112] reduct computation.

4.2 Preliminaries

This section present the basic notions about rough set theory and fuzzy rough set theory with its corresponding feature subset selection methods via discernibility matrix that are useful in understanding our proposed work.

4.2.1 Rough Set Theory

Rough set theory (RST), proposed by Z. Pawlak in the early 1980s, is a mathematical framework for describing and concisely exploiting data dependencies from a domain without any need of prior and external knowledge about data [76, 77, 78]. In particular, a rough set approach serves very well in the direction of reduction of superfluous attributes preserving the same knowledge as given by the full set of attributes.

Let $DT = (U, C^c \cup \{d\}, \{V_a, f_a \mid a \in C^c\}, \{V_d, f_d\})$ be the decision system, where $U = \{x_1, x_2, \ldots, x_n\}$ is a non-empty finite set of objects (universe of discourse), $C^c = \{a_1, a_2, \ldots, a_m\}$ is a non-empty finite set of categorical conditional attributes. $\{d\}$ is a distinguished attribute or decision attribute such that $C^c \cap \{d\} = \emptyset$. V_a is the set of conditional attribute values of 'a', and f_a is an information mapping from U to V_a i.e., $f_a : U \to V_a$. V_d is the set of decision attribute values (decision categories), and f_d is an information mapping from U to V_d i.e., $f_d : U \to V_d$.

Given a decision system DT with any subset $P \subseteq C^c$, there is an associated equivalence (indiscernibility) relation $IND(P)$ defined over $U \times U$, and defined as follows:

$$IND(P) = \{(x_1, x_2) \in U \times U \mid \forall a \in P, f_a(x_1) = f_a(x_2)\} \tag{4.1}$$

where, $f_a(x)$ represents the value of object x on attribute a. $(x_1, x_2) \in IND(P)$ denotes that x and y are indiscernible by attributes from P means, they have same vectors of attribute values for attributes in P. The equivalence relation $IND(P)$ partitions the universe U into a family of disjoint subsets, which are the set of equivalence classes generated by $IND(P)$. The family of all equivalence classes of the relation $IND(P)$ are represented as $U/IND(P)$, or U/P. In particular, U/D denotes the set of decision equivalence classes. An equivalence class of any object $x \in U$ is represented as $[x]_P$ and defined as: $[x]_P = \{y \in U \mid (x, y) \in IND(P)\}$.

A subset of features selected using RST is named as *reduct*, and the process is called reduct computation (feature subset selection). Reduct is defined as a minimal subset of conditional attributes preserving the classifying ability of the original decision system [45]. There are two important procedures/approaches to finding rough set reducts: Degree of Dependency and Discernibility Matrix.

Our proposed work is based on a discernibility matrix construction. So, we are only providing the preliminaries about discernibility matrix based reduct computation.

4.2.1.1 Decision-Relative Crisp Discernibility Matrix

Skowron and Rauszer [103, 122] introduced the concept of crisp discernibility matrix (DM) for finding rough reducts of a given decision system. Crisp DM is a representation for crisp discernibility relation. In discernibility relation (a complement of indiscernibility relation), two objects can be discernible in a given decision system if their values differ in at least one attribute. Given $P \subseteq C^c$, a discernibility relation on P is denoted as $DISC(P)$ where a pair of objects x and y in U belong to $DISC(P)$ if and only if there exists at least one attribute in P having different values for x and y.

$$DISC(P) = \{(x, y) \in U \times U \mid \exists a \in P, a(x) \neq a(y)\} \tag{4.2}$$

A discernibility matrix of DT is a symmetric matrix of order $|U| \times |U|$ i.e., $M(x, x) = \emptyset$ and $M(x, y) = M(y, x)$. Hence we consider only the lower or the upper triangular of the matrix. DM stores the sets of conditional attributes that can discern pairwise comparison of all objects from U.

The objective of finding reducts is more interesting when considering only those object pair discernibility when their corresponding decision attribute differ, called a decision-relative discernibility matrix. Each entry is defined in Eqn. (4.3).

$$M(x, y) = \begin{cases} \{a \mid a \in C^c,\ f_a(x) \neq f_a(y)\}, & \text{if } f_d(x) \neq f_d(y) \\ \emptyset, & \text{otherwise} \end{cases} \tag{4.3}$$

Each entry $M(x, y)$ consists of those conditional attributes that differentiate object pair x and y.

From this, the discernibility function for a given decision system DT can be introduced. A discernibility function $f(\{d\})$ is a boolean function of the discernibility matrix (M) and can be defined as:

$$f(\{d\}) = \wedge\{\ \vee M(x, y) \mid \forall (x, y) \in U \times U, M(x, y) \neq \emptyset\} \tag{4.4}$$

The expression $\vee M(x, y)$ is the disjunction of all conditional attributes in $M(x, y)$, implying that the object pair can be distinguished by any attribute of $M(x, y)$. The expression $\wedge\{\vee M(x, y)\}$ is the conjunction of all $\vee M(x, y)$, implies that all pairs of objects of different decision classes need to be discerned.

Finding the *reducts* of the decision system DT is equivalent to the problem of transforming the discernibility function (conjunctive normal form) into reduced logical expression disjunctive normal forms (without negation) using absorption and distribution law. The logical expression of each conjunction of the reduced disjunctive form is known as a prime implicant.

If a set of attributes set $P \subseteq C$ is a reduct if and only if the conjunction of all attributes in P is a prime implicant of $f(\{d\})$. Hence, finding the set of all individual prime implicants of the discernibility function provides all minimal solutions to the boolean function. Although this is guaranteed to find all reducts of DT, still it is an *NP-Hard* approach [80, 103, 122].

Skowron and Rauszer [103] also proposed different characterization of a reduct P based on DM. Given a decision system DT, a set of conditional attributes P is said to be *reduct* if and only if:

Property 1: $\forall(x, y) \in U \times U : [M(x, y) \neq \emptyset \Rightarrow P \cap M(x, y) \neq \emptyset]$

Property 2: $\forall a \in P,\ \exists(x, y) \in U \times U : [M(x, y) \neq \emptyset \wedge ((P - \{a\}) \cap M(x, y) = \emptyset)]$

Property 1 presents that reduct R is sufficient to distinguish all discernible objects pairs means, every entry of DM holds property 1. Property 2 establishes that each attribute in reduct P is important and indispensable. Both properties provide a sufficient way to examine whether a resulted subset of attributes is reduct or not. However, many researchers have developed several efficient heuristic algorithms based on DM for reduct computation [45, 144]. Out of them, the Johnson Reducer strategy is one of the popular approaches widely used for reduct computation [45].

4.2.2 Fuzzy Rough Set Theory

The formulation of classical Pawlak's rough sets [77] can only operate effectively with datasets containing symbolic (qualitative) attributes where indiscernibility relation plays an important role. Application of classical rough sets to real-valued attributes will produce feature subsets with finer granularity. Hence, the induced rules from the selected features suffer from poor generalizability to test datasets.

So, one of the solutions is to discretize the dataset beforehand and produce a new dataset with categorical values [68, 71]. However, the discretization method is often inadequate and causes essential information loss that can hamper subsequent feature subset selection quality and result in significant misclassification in pattern space. Even finding an optimal way for the discretization process in a dataset is an NP-Hard problem [70].

Dubois and Prade [22, 23] introduce the constructive approach called fuzzy rough sets (FRS) in the early 1990s to combine the coarseness of rough sets [77] with the vagueness of fuzzy sets [126] to operate on hybrid decision systems.

Let $HDT = (U, C^h = (C^c \cup C^n) \cup \{d\}, \{V_{a_c}, f_{a_c}\}_{a_c \in C^c \cup \{d\}}, \{V_{a_n}, f_{a_n}\}_{a_n \in C^n})$ be the hybrid decision system, where U is the finite set of universe, C^c is the categorical/qualitative conditional attributes, C^n is the numerical/quantitative conditional attributes, C^h constitutes hybrid conditional attributes C^c and C^n along with discrete decision attribute d. V_{a_c} is a finite domain value set of attribute $a_c \in C^c \cup \{d\}$ and f_{a_c} is a mapping of assigning a symbol from U to value set V_{a_c} i.e., $f_{a_c} : U \rightarrow V_{a_c}$. V_{a_n} is a finite set of domain values having real-valued attribute a_n with range of interval V_a^n s.t. $a_n \in C^n$ and f_{a_n} is a mapping assigning a value from universe U to value set V_{a_n} i.e., $f_{a_n} : U \rightarrow V_{a_n}$. Notation, $a_c(x)$ and $a_n(x)$ are used in the place of the symbol $f_{a_c}(x)$ and $f_{a_n}(x)$ for simplicity and better readability.

Fuzzy rough set theory extends the notion of rough equivalence relation [77, 78] into an idea of fuzzy similarity/equivalence relation or a fuzzy tolerance relation, resulting in a fuzzy partition of the universe U. The key concept of FRS is fuzzy similarity relation. Fuzzy similarity relation determines the degree to which any two objects $(x, y) \in U \times U$ are similar in U for given quantitative attribute values.

Fuzzy similarity relation μ_{R_a} on $U \times U$ on attribute $a \in C^h$ satisfying the following requirements is called as fuzzy tolerance relation, if

$$\mu_{R_a}(x, x) = 1 \; \forall x \in U \tag{4.5}$$

$$\mu_{R_a}(x, y) = \mu_{R_a}(y, x) \; \forall \; x, y \in U \tag{4.6}$$

$$\mu_{R_a}(x, z) \geq \Gamma(\mu_{R_a}(x, y), \mu_{R_a}(y, z)) \; \forall \; x, y, z \in U \tag{4.7}$$

Eqn. (4.5) and Eqn. (4.6) hold the reflexivity and symmetry properties of equivalence relation, while Eqn. (4.7) renders additional requirement of Γ-transitivity i.e., given a t-norm Γ. A triangular norm (t-norm) Γ is an associative binary operator on the interval $[0, 1]$ holding increasing, monotonic, commutative and associative property with $[0, 1]^2 \rightarrow [0, 1]$ information mapping satisfying boundary condition $\Gamma(1, x) = x, \forall x \in [0, 1]$ [5, 37]. Hence, μ_{R_a} is also called as fuzzy Γ-equivalence relations to represent the approximate equality.

Given a hybrid decision system HDT, the fuzzy equivalence relation with respect to each numeric attribute a_n ($\forall a_n \in C^n$) is defined as $\mu_{R_{a_n}}$, where $\mu_{R_{a_n}}(x, y)$ represent the degree of similarity between x and y for attribute values of a_n. To express the fuzzy similarity relation $\mu_{R_{a_n}}$ between two objects $(x, y) \in U$ w.r.t. 'a_n' attribute, there are some widely used examples of fuzzy relations $\mu_{R_{a_n}}$ for this purpose, such as [41]:

$$\mu_{R_{a_n}}(x, y) = exp\left(-\frac{(a_n(x) - a_n(y))^2}{2\sigma_{a_n}^2}\right) \tag{4.8}$$

$$\mu_{R_{a_n}}(x, y) = max\left(min\left(\frac{(a_n(y) - (a_n(x) - \sigma_{a_n}))}{(a_n(x) - (a_n(x) - \sigma_{a_n}))}, \frac{(a_n(x) + \sigma_{a_n}) - a_n(y))}{(a_n(x) + \sigma_{a_n}) - a_n(x))}\right), 0\right) \tag{4.9}$$

where, σ_{a_n} is the standard deviation for a_n attribute, $\sigma^2_{a_n}$ is the variance of a_n attribute and $a_n(x)$ represent object value on attribute a_n.

In particular for qualitative attributes a_c, fuzzy equivalence relation is considered to be crisp equivalence relation based on indiscernibility relation and thus defined as,

$$\mu_{R_{a_c}} = \begin{cases} 1 & if \quad a_c(x) = a_c(y) \\ 0 & if \quad a_c(x) \neq a_c(y) \end{cases} \tag{4.10}$$

Given a HDT, a fuzzy similarity relation is expanded to a subset of attributes $P \subseteq C^h$ using t-norm (Γ).

$$\mu_{R_P}(x,y) = \Gamma \underbrace{(\mu_{R_a}(x,y))}_{a \in P} \ \forall x, y \in U \tag{4.11}$$

4.2.2.1 Decision-Relative Fuzzy Discernibility Matrix

The development of crisp DM [103, 122] is often used in RST for reduct computation may also be extended in fuzzy rough reduct computation. Jensen et al. [41] proposed an extension of crisp DM to fuzzy case for use in FRS model to determine fuzzy-rough reducts. Crisp DM is extended to Fuzzy DM by considering fuzzy clauses as discernibility entries instead of crisp clauses [41]. Each entry (known as clause) corresponds to fuzzy DM is a fuzzy set over attribute space containing the discernibility value of each attribute [41].

Let $\mu_{R_a}(x,y)$ is a fuzzy similarity relation between objects x and y on an attribute $a \in C^h$. A fuzzy discernibility measure $(\mu_{DR_a}(x,y))$ w.r.t. attribute 'a' is obtained by performing fuzzy negation on $(\mu_{R_a}(x,y))$.

$$\mu_{DR_a}(x,y) = Neg(\mu_{R_a}(x,y)) \ x, y \in U \tag{4.12}$$

where Neg is a fuzzy negator, and $\mu_{DR_a}(x,y)$ is a degree of the fuzzy discernibility of objects x and y w.r.t attribute 'a' ($a \in C^h$). A fuzzy negator Neg is a decreasing $[0,1] \to [0,1]$ mapping that satisfies $Neg(1) = 0$ and $Neg(0) = 1$ for all x in $[0,1]$. The standard negation is defined as $Neg(x) = 1 - x$ and the same is used in our work.

For a crisp case, the resulting relation is determined to be $\mu_{DR_a}(x,y) = 1$ (when objects are discernible w.r.t. attribute 'a') and $\mu_{DR_a}(x,y) = 0$ (when objects are indiscernible w.r.t. attribute 'a'). For a fuzzy case, the respective value for $\mu_{DR_a}(x,y)$ is in range of $[0,1]$, providing a graded discernible measure.

Given a *HDT*, each entry (or clause) $M(x,y)$ in the fuzzy DM contains a set of all conditional attributes of size $|C^h|$ associated with their discern membership/degree for objects x

and y. To a given decision system, only those object pairs entries with different classes are included in fuzzy DM [41], called a decision-relative fuzzy DM. Each entry is defined as:

$$M(x,y) = \begin{cases} \{a_s \mid a \in C^h, s = Neg(\mu_{R_a}(x,y))\}, & \text{if } f_d(x) \neq f_d(y) \\ \emptyset, & \text{otherwise} \end{cases} \tag{4.13}$$

For example, $M(x,y)$ might be $\{a_{0.3}, b_{0.5}, c_{0.9}\}$. Here, $a_{0.3}$ indicates $\mu_{DR_a}(x,y) = 0.3$. The fuzzy discernibility relation is stored in $|U| \times |U|$ a symmetric matrix with each entry as an array of discernibility values.

As similar to the crisp discernibility function, the discernibility function $f(\{d\})$ is encoded in fuzzy version once the decision-relative fuzzy DM is constructed.

$$f(\{d\}) = \{\wedge\{\ \vee M(x,y)\} \leftarrow d_{(Neg(\mu_{R_d}(x,y)))}\}\ \forall\ (x,y) \in U \times U \tag{4.14}$$

$\leftarrow$ represents fuzzy implication. The discernibility function returns value in range of $[0,1]$. Only clauses with different decision values are included in fuzzy DM. The different decision values affect the overall satisfiability of the clauses largely. Reducts are calculated via fuzzy intersection of all clauses from the construction of fuzzy discernibility function may not render sufficient information to evaluate subsets [41]. So, considering the individual satisfiability of each clause for a given set of attributes provide more information to evaluate subsets.

The degree of satisfaction of a clause $M(x,y)$ for a given subset of attributes P ($P \in C^h$) with respect to the decision attribute $\{d\}$ is defined as:

$$SAT_{P,\{d\}}(M(x,y)) = \underset{a \in P}{S}\{M^a(x,y)\} \tag{4.15}$$

where S is a t-conorm, and $M^a(x,y)$ is a degree of satisfaction of a clause w.r.t. attribute ‘a’. The dual notion to a t-norm is a t-conorm, where its neutral element is 0 instead of 1 [37]. A triangular conorm (t-conorm) S is a binary operator on the interval $[0,1]$ holding monotonic, commutative and associative property with $[0,1]^2 \rightarrow [0,1]$ information mapping satisfying boundary condition $S(x,0) = x, \forall x \in [0,1]$ [37].

In crisp propositional satisfiability, each clause has been completely satisfied if at least one variable in the clause is set to true. For fuzzy cases, each clause has been satisfied when it reaches to maximum satisfiability degree.

Based on Eqn. (4.15), the total satisfiability of entire clauses for a subset $P \in C^h$ can be calculated as:

$$SAT(P) = \frac{\sum\limits_{x,y \in U, x \neq y} SAT_{P,\{d\}}(M(x,y))}{\sum\limits_{x,y \in U, x \neq y} SAT_{C^h,\{d\}}(M(x,y))}\ \forall\ x,y \in U \tag{4.16}$$

A minimal subset of conditional attributes $P \subseteq C^h$ is referred as a fuzzy rough reduct, if and only if the following condition satisfies:

1. $SAT(P) = SAT(C^h) = 1$ (Jointly Sufficient Condition)
2. $\forall P' \subset P, SAT(P') < SAT(P)$ (Individually Necessary Condition)

Property 1 shows that given a decision system, the jointly sufficient condition states that the satisfiability measure of reduct (P) is collectively sufficient to induce the same satisfiability measure of all conditional attribute (C^h). Property 2 shows that the individually necessary condition states that none of the reduct attributes can be omitted as each of them is necessary.

4.3 Literature Review of Fuzzy Rough Set Theory

The traditional FRS approaches are proved very popular for feature subset selection. The first pioneering work in fuzzy rough feature selection (FRFS) is presented by Jensen et al. [38] using Dubois-Prade's fuzzy rough set model. It performed well in terms of retaining fewer attributes with higher classification accuracy than RST based reduction on web dataset, which aided in web categorization. In [38], the authors proposed an algorithm to compute a close-to-minimal reduct based on dependency function and also measure the quality of attributes. Subsequently, several aspects of improvement based on features selection [39, 40, 95] and computation time were done for [38].

In [41], the authors introduced three robust techniques based on the fuzzy similarity relation, which overcame the problems in papers [38, 40] and also developed the fuzzy DM for computing the feature selection. In particular, these techniques have shown high flexibility and reduced the complexity of computing the cartesian product of fuzzy equivalence classes in [38, 40]. This approach [41] received the several considerations of researchers in [10, 12, 13, 15, 45, 83, 92] and became an effective approach for reduct computation.

Standard FRS approaches consider every data object compared with every other object of different classes in generating the fuzzy similarity relations for calculating the dependency measure and constructing a DM. These approaches show scalability issues for large datasets because they consider all objects contained in the data while generating fuzzy similarity relations. So, each data object is compared with every other object for inducing fuzzy similarity relations. This calculation requires $O(n^2)$ comparisons (where n is the number of data objects). Thus, the memory utilization for constructing similarity matrices is $O(|U|^2|C^h|)$, where $|U|$ is the size of the object space and $|C^h|$ is the size of the attribute space. An increase in data size will have a negative impact on runtime on these FRS approaches. These algorithms face problems from both data storage and computational complexity viewpoints. Several attempts have been developed in literature in developing a scalable approach for FRS reduct computation [15, 43, 113, 138]. These approaches primarily aim to reduce the requisite space complexity. Some examples in this direction are representative instance-based approaches [138], accelerating positive region [72], transforming fuzzy DM into crisp DM [15, 44].

To mitigate the processing overhead on FRS approaches, In 2015, Jensen et al. [43] presented two approaches to FRS intending to reduce the computational complexity in reduct computation on large datasets. The first approach (nnFDM) is to compute the membership degree of each object with k-nearest neighbour objects of different decision classes in both calculating dependency measures and constructing fuzzy DM. The second approach is to create a set of groups of features through correlation and then use the fuzzy–rough dependency measure to discover good subsets and then choose the top-ranked feature from each discovered group. After selecting features, the process of creating groups is iterated, avoiding earlier chosen features. This process repeats until the stopping criterion is reached. Although these two ideas are tackling the problem of computation associated with large data, their performance is also being affected [43].

In 2015, Wang et al. [113] introduced a fitting model for the classical FRS model (NFRS) for overcoming the problem of overfitting by reduct, resulting in misclassification, especially in datasets with high overlapping across different categories. The idea is to compute a fuzzy decision of a sample using the concept of fuzzy neighborhood that can fit a given sample and guarantee to determine maximal membership degree on its own category, which effectively prevents classification error.

In 2018, Zhang et al. [138] developed an FRS based feature selection approach (FWARA) using representative instances to alleviate the computational complexity through minimal knowledge. The objective is to determine the representative instances as minimal knowledge that can cover the same decision discrimination ability as compared to all objects to induce all the fuzzy granular rules. Then, a fuzzy dependency function is formulated to compute feature subset selection using representative instances. Furthermore, a wrapper strategy is applied to selected features subset to find the best quality feature subset that achieves classification ability.

In 2018, Dai et al. [15] presented two different diverse approaches (RMDPS and WR-MDPS) for FRS reduct computation with the concern of reduced maximal discernibility pairs in fuzzy DM construction. They follow the idea of transforming the notion of fuzzy DM into crisp DM construction to consider those pairwise comparison of objects that can have maximal discernible attributes or minimal fuzzy similarity attributes.

In 2019, Peng et al. [72] proposed an accelerator based on the positive region in the process of feature selection (PARA). The author's idea is to keep only discernible objects which can update the positive region to avoid redundant computation and accelerate attribute reduction.

4.4 Motivation

The above-mentioned scalable FRS approaches, i.e., nnFDM [43], NFRS [113], FWARA [138], RMDPS [15], WRMDPS [15] and PARA [72], achieved significant scalability against traditional FRS approaches. However, they still have some limitations that they could not compute to such an extent on large datasets. nnFDM approach requires nearest neighbour calculation for each object prior to computing which is a costly task. Both FWARA and PARA require the generation of fuzzy similarity matrices having a memory requirement of $O(|U|^2|C^h|)$ beforehand to select representative instances where $|U|$ is the size of the object space and $|C^h|$ is the size of the attribute space. RMDPS and WRMDPS require $O(|U|^2|C^h|)$ memory space priorly for DM construction as a pairwise comparison of every object against every object that belongs to different classes. An increase in object space would have adverse impacts upon computational overhead in these approaches. They are also preserving the information of every object, which may lead to the fact that these reduction algorithms select more features and consume more computational time.

The objective of the thesis is to reduce the space complexity using FMNN as a granular computing technique (as described in Chapter 2) for enhancing the scalability of FRS feature subset selection approach. In this chapter, we explored how to achieve this objective of achieving better scalability in reduct computation without sacrificing model accuracy using FMNN as preprocessing step. The knowledge of FMNN in terms of hyperboxes can decrease the space complexity and the computation time required in FRS-based reduct computation. As FMNN is ordered-dependent, the obtained FRS reduct through FMNN preprocessing can vary based on the order in which the data is presented to FMNN. As the objective of FRS reduct computation algorithm is only to compute one reduct out of all possible reducts existing in the given decision system, order dependency of FMNN will not become a barrier.

In this chapter, such an intuitive idea is introduced for a solution to FRS feature subset selection by using the concept of hyperbox utilizing FMNN [102] as a preprocessor.

Note: *Due to the nature of FMNN, the proposed approach works only on numeric decision systems. Let* $DT = (U, C^n \cup \{d\}, \{V_a, f_a \mid a \in C^n\}, \{V_d, f_d\})$ *be the decision system with numeric conditional attributes and in the remaining part of the thesis, DT refers to numeric decision system.*

4.5 Proposed FDM-FMFRS Reduct Algorithm

In this section we propose a fuzzy DM based fuzzy rough reduct computation algorithm named as FDM-FMFRS (FDM: Fuzzy discernibility matrix, FM: Fuzzy min-max neural network, FRS: Fuzzy rough set). We propose a novel approach to increase the scalability of the FRS approach by constructing the granular model from object space before applying it to the FRS model. This granular model is designed by collecting information granules regarded as hyperboxes using FMNN [102], described in Section 2.2, Chapter 2.

This chapter aims to compute an approximate reduct efficiently with the advantage in space and time complexity. The concept of the approximate reduct is introduced by Slezak [104] that contains the potential attributes to achieving near to exact reduct capability. Even, the above-mentioned approaches [43, 72, 83, 138] also result in an approximate reduct. In FDM-FMFRS, we introduce a solution for FRS reduct computation, utilizing the FMNN learning as a preprocessor step. The proposed work FDM-FMFRS is summarized as follows:

1. Creation of interval-valued decision system from FMNN preprocessing.

2. Fuzzy discernibility matrix construction based on interval-valued decision system.

3. Find an approximate reduct computation based on fuzzy discernibility matrix.

4.5.1 Creation of Interval-Valued Decision System from FMNN

The traditional FMNN algorithm, as described in Section 2.2, has a three-step learning procedure such as expansion, overlap, and contraction for each input pattern. The overlapping and contraction steps result in non-overlapping between pairs of hyperboxes belonging to different decision classes. This disambiguation helps in crisp decision-making for classification but results in crucial information loss of boundary regions between decision classes.

In this chapter, our objective of FMNN preprocessor is to aid in the construction of fuzzy DM. But, following the traditional procedure of FMNN may lose valuable information to represent discernibility among the objects of different classes. Hence, we have considered a simplified FMNN training procedure to restrict only the expansion step for preserving the naturally overlapping regions among the hyperboxes of multiple decision classes. We have incorporated the proposed kNN-FMNN training phase, as described in the Algorithm [3.1] (Section 3.3.1), for the proposed FDM-FMFRS.

Algorithm [3.1] gives the simplified FMNN training process for arriving at hyperboxes with possible overlap among multiple decision classes. FMNN preprocessing results in hyperboxes where each hyperbox represents granule of objects of a decision class. The set of objects of hyperbox is the objects having absolute membership of one. As the objects are in the nearby vicinity, there are chances that most of them represent a single decision class; still, some exceptions can exist as overlapping among hyperboxes can't be avoided as described in Section 3.3.1.

Here, we construct the interval-valued decision system (IDS) based on hyperboxes, resulting from the training of kNN-FMNN (as described in Chapter 3) on the given decision system DT. IDS can retain the boundary information of overlapping intervals to each attribute in the decision system. The hyperbox is bounded by V (minimum point), and W (maximum point) represents the area in space belonging to a particular decision class. This representative hyperbox is taken as a single entity for representing the member objects and becomes an object in the resulting IDS.

Let IDS $= (HBS, C^n \cup \{d\})$ be interval-valued decision system, where $HBS = \{H_1, H_2, \ldots, H_r\}$ represents the universe of hyperboxes. Let $[V^H, W^H]$ represent the minimum and maximum points of hyperbox H. In IDS, the value of a hyperbox $H \in HBS$ over an attribute $a \in C^n$ is represented by the interval v_a^H to w_a^H ($[v_a^H, w_a^H]$), where, v_a^H is component of minimum point V^H and w_a^H is component of maximum point W^H corresponding to the attribute a. The value of decision attribute d is taken as per the decision class to which H belongs.

4.5.2 Fuzzy Discernibility Matrix Construction based on Interval-Valued Decision System

In this section, we are constructing the fuzzy DM based on IDS, as obtained in Section 4.5.1. Each clause in the fuzzy DM corresponds to a pair of hyperboxes representing different classes. Based on Eqn. (4.13), an entry of fuzzy DM corresponding to hyperboxes H_i and H_j is a vector of fuzzy discernibility measure for all attributes. In fuzzy DM construction for the decision system, the valid entries are defined as a pair of hyperboxes belonging to different decision classes. To find a fuzzy discernibility measure, we require a fuzzy similarity measure applicable to interval-valued data. Several similarity measures are defined in the literature for interval-valued data [36, 48]. Out of these, Jaccard's similarity measure (JS) [36] is used for the proposed work.

Jaccard's similarity measure [36] introduces the concept of similarity measure for interval-valued data based on real numbers. It satisfies the boundness, symmetry, reflexivity, and transitivity properties of a similarity measure. Hence, Jaccard's similarity is a fuzzy equivalence relation defined over the universe of interval-valued data objects. Let I_x and I_y represent two overlapping intervals. The Jaccard's similarity measure $JS(I_x, I_y)$ is defined as:

Algorithm 4.1: Creating Fuzzy Discernibility Matrix

Input : HBS: Set of hyperboxes, C^n: Set of conditional attributes
Output: M: Fuzzy Discernibility Matrix.

for *every* H_i *in* $|HBS|$ **do**
  for *every* H_j *in* $|HBS|$ **do**
    // Compute $M(H_i, H_j)$ for i^{th} hyperbox with each j^{th} hyperbox of different class labels
    if $d(H_i) \neq d(H_j)$ **then**
      for *each* a *in* C^n **do**
        $M^a(H_i, H_j) = Neg(JS([v_a^{H_i}, w_a^{H_i}], [v_a^{H_j}, w_a^{H_j}]))$ from Eqn. (4.19)
      end
    end
  end
end
return M

$$JS(I_x, I_y) = \frac{|I_x \cap I_y|}{|I_x \cap I_y| + |I_x \setminus I_y| + |I_y \setminus I_x|} \tag{4.17}$$

where, $|I_x \cap I_y|$ is the size of intersection between I_x and I_y. $|I_x \setminus I_y|$ is the size of the interval segment of I_x that are not overlapping with I_y. Similarly, $|I_y \setminus I_x|$ is the size of the interval segment of I_y that are not overlapping with I_x. If I_x and I_y do not overlap, then $JS(I_x, I_y) = 0$ indicates both intervals are completely different from each other. Likewise, if I_x and I_y are fully overlapping, then $JS(I_x, I_y) = 1$ indicates both intervals are completely identical.

Using JS, the fuzzy DM entry between H_i and H_j, denoted as $M(H_i, H_j)$, belonging to different classes is defined as:

$$M(H_i, H_j) = \begin{cases} \{a_s \mid \forall a \in C^n, s = Neg(J_S([v_a^{H_i}, w_a^{H_i}], [v_a^{H_j}, w_a^{H_j}]))\}, & \text{if } \mathrm{d}(H_i) \neq \mathrm{d}(H_j) \\ \emptyset, & \text{otherwise} \end{cases} \tag{4.18}$$

The component corresponding to attribute $a \in C^n$ is:

$$M^a(H_i, H_j) = Neg(JS([v_a^{H_i}, w_a^{H_i}], [v_a^{H_j}, w_a^{H_j}])) \tag{4.19}$$

where, Neg denotes the fuzzy negation and we have used standard negation i.e., Neg(x) = 1-x in our implementation.

Algorithm [4.1] presents the structure for computing the fuzzy DM based on IDS. In Algorithm [4.1], for every pair of hyperboxes of different classes, an entry $M(H_i, H_j)$ corresponds to fuzzy discernibility measure for all attributes between H_i and H_j, based on Eqn. (4.18).

Fuzzy DM constructed in this manner is an approximation to fuzzy DM constructed at an object level. So, one can say that a pair of hyperboxes comparison absorbs many pairs of objects based comparison. The cardinality of hyperboxes ($|HBS|$) is usually much lesser than the cardinality of objects ($|U|$) i.e., ($|HBS| << |U|$). A fuzzy DM entry between a pair of objects of different classes is always a superset of the corresponding fuzzy discernibility matrix entry between the hyperboxes containing these objects. Hence, the fuzzy rough reduct computed using fuzzy DM for IDS results as an approximate reduct. Hence, validating the quality of the approximate reduct becomes the important objective in our experimental studies described in Section 4.7.

4.5.3 Approximate Reduct Computation based on Fuzzy Discernibility Matrix

Algorithm 4.2: Finding an Approximate Reduct

Input : FDM :Fuzzy discernibility matrix, C^n: Conditional attributes
Output: Red: Approximate Reduct

$Red = \emptyset$;
$SAT(Red) = 0$, $SAT(C^n) = 1$;
while $SAT(Red) \neq SAT(C^n)$ **do**
    $a^{sat} = 0$, $a^{best} = \emptyset$;
    for *each* $a \in C^n - Red$ **do**
        $S_a = SAT(Red \cup \{a\})$;
        if $S_a > a^{sat}$ **then**
            $a^{sat} = S_a$;
            $a^{best} = \{\text{a}\}$;
        end
    end
    Red = Red $\cup \{a^{best}\}$;
end
return *Red*

In this section, we provide an approximate reduct computation algorithm using fuzzy DM constructed on IDS as given in the Section 4.5.2. Algorithm 4.2 gives the procedure for computing an approximate reduct based on fuzzy DM. The satisfiability measure with Lukasiewicz t-conorm (S(x,y) = $min\{1, x + y\}$) is considered [15] to calculate individual satisfaction of each clause over attributes. The Algorithm [4.2] follows the sequential forward selection (SFS) control strategy. Algorithm starts with reduct *Red* initialize to an empty set. In each iteration, SAT measure is computed using Eqn. (4.16) for each attribute (($Red \cup$

$\{a\}) \ \forall a \in C^n - Red)$ not already included in Red. The attribute achieving maximum SAT measure (a^{best}) is included in the reduct set Red. The algorithm terminates when $SAT(Red)$ becomes equal to $SAT(C^n)$ (i.e., 1) and returns the obtained approximate reduct Red.

Here, the entire motivation behind this work is to use an interval-valued decision system instead of an object space decision system in Fuzzy DM construction for reduct computation that can significantly decrease computational time and memory utilization.

4.6 Complexity Analysis of FDM-FMFRS Algorithm

This section shows the time and space complexity analysis of the proposed algorithm FDM-FMFRS. The following variables are used in the complexity analysis of FDM-FMFRS.

- $|U|$: the number of objects.
- $|HBS|$: the number of hyperboxes.
- $|C^n|$: the number of numeric conditional attribute.
- $|M|$: Size of discernibility matrix

Table 4.1 shows the time complexity of the proposed algorithm FDM-FMFRS. In Table 4.1, Algorithm 3.1 corresponds to the construction of IDS, whose time complexity was discussed in Chapter 3 (Section 3.4) and had a time complexity of $O(|U| * |HBS| * |C^n|)$. In Table 4.1, Algorithm 4.1 with steps 1-9 constructs the fuzzy DM based on IDS given in Algorithm 3.1 with a time complexity $O(|HBS|^2 * |C^n|)$. Algorithm 4.2 with steps 3-13 perform reduct computation on fuzzy DM using SFS based control strategy with a time complexity of $O(|M| * |C^n|^2) = O(|HBS|^2 * |C^n|^2)$, since $|M| = O(|HBS|^2)$.

So, the total complexity of the proposed algorithm FDM-FMFRS is: $O(|U| * |HBS| * |C^n|) + O(|HBS|^2 * |C^n|^2)$.

The space requirement of FDM-FMFRS is for three sources: First, the decision system is required for constructing IDS with a space complexity of $O(|U| * |C^n|)$. Second, IDS-based fuzzy DM is constructed with a requirement of space complexity $O(|HBS| * |C^n|)$. Finally, the fuzzy DM is required for generating the reduct having a space complexity $O(|M| * |C^n|) = O(|HBS|^2|C^n|)$. Thus, the total space complexity of FDM-FMFRS algorithm is $O(|U| * |C^n|) + O(|HBS|^2 * |C^n|)$.

Table 4.1: Time Complexity Analysis of FDM-FMFRS Algorithm

Algorithm (phase)	Steps in Algorithm	Time complexity
Algorithm 3.1	2-29. Construction of IDS	$O(\|U\| * \|HBS\| * \|C^n\|)$
Algorithm 4.1	1-9. Construction of fuzzy DM	$O(\|HBS\|^2 * \|C^n\|)$
Algorithm 4.2	3-13. Reduct computation	$O(\|M\| * \|C^n\|^2) = O(\|HBS\|^2 * \|C^n\|^2)$

Table 4.2: Benchmark Datasets

Dataset	Attributes	Objects	Class
Ionosphere	32	351	2
Vehicle	18	846	4
Segment	16	2310	2
Steel	27	1941	7
Ozone Layer	72	1848	2
Page	10	5472	5
Robot	24	5456	4
Waveform2	40	5000	3
Texture	40	5500	11
Gamma	10	19020	2
Satimage	36	6435	6
Ring	20	7400	2
Musk2	166	6598	2
Shuttle	9	57999	7
Sensorless	48	58509	11
MiniBooNE	50	129596	2
Winnipeg	174	325834	7

4.7 Experiments and Results

The hardware configuration of the system used for experiments is CPU: Intel(R) i7-8500, Clock Speed: 3.40GHz × 6, RAM: 32 GB DDR4, OS: Ubuntu 18.04 LTS 64 bit and Software: Matlab R2017a. The detailed experimental evaluation is conducted on seventeen benchmark numeric decision systems taken from UCI machine learning repository [21], the details are given in Table 4.2. The proposed algorithm FDM-FMFRS is implemented in the Matlab environment. In our experiments, we set the sensitive parameter γ value equal to 4, as recommended [63, 102]. And, based on the selected theta (θ) parameter in Chapter 3, we deduced that *theta* value of 0.3 are appropriate in computation FDM-FMFRS reduct. In FDM-FMFRS experiment, Lukasiewicz t-conorm ($S(x, y) = min\{x + y, 1\}$) for Eqn. (4.15) and fuzzy standard negation ($Neg(x) = 1 - x$) Eqn. (4.17) are used.

The performance of the proposed algorithm FDM-FMFRS is assessed by comparing it with recent state-of-the-art approaches developed for FRS reduct computation in 2018 and 2019, named as RMDPS [15], WRMDPS [15], FWARA [138] and PARA [72]. FWARA and PARA codes are provided by their corresponding author in Matlab environment, and RMDPS and WRMDPS codes are implemented by my supervisor in Matlab environment. Furthermore, these comparative approaches (RMDPS, WRMDPS, FWARA and PARA) follow their own fuzzy model with t-norm, t-conorm and fuzzy similarity relations as given in the respective publications and experiments are conducted in the same environment stated above. The comparative experiments are conducted in the same system using Matlab environment. The performance of FDM-FMFRS is examined through a comparative evaluation with respect to the following objectives:

1. Evaluate quality of approximate reduct through Gamma measure.

2. Comparative analysis of proposed approach in construction of different classifiers through ten-fold cross-validation (10-FCV).

Table 4.3: Relevance of FDM-FMFRS reduct through Gamma measure

Datsets	**Gamma Meausre**					
	UNRED	**FDM-FMFRS**	**RMDPS**	**WRMDPS**	**FRAWA**	**PARA**
Ionosphere	0.99	0.98	0.99	0.99	0.99	0.99
Segment	0.98	0.94	0.98	0.98	0.98	0.96
Steel	0.98	0.94	0.98	0.98	0.98	0.98
Vehicle	0.99	0.99	0.99	0.99	0.99	0.99
Ozone	1	0.99	1	1	1	1
Page	0.87	0.85	0.87	0.87	0.87	0.87
Texture	0.99	0.94	0.99	0.99	0.99	0.93
Waveform2	1	1	1	1	1	1
Robot	0.97	0.90	0.97	0.97	0.97	0.97
Satimage	0.99	0.98	0.99	0.99	0.99	0.98
Ring	1	1	1	1	1	1
Datsets	**Reduct Length**					
	UNRED	**FDM-FMFRS**	**RMDPS**	**WRMDPS**	**FRAWA**	**PARA**
Ionosphere	32	7	27	27	31	18
Segment	16	9	15	15	14	10
Steel	27	11	21	21	18	15
Vehicle	18	15	18	18	17	14
Ozone	72	9	39	42	54	29
Page	10	8	10	10	10	9
Texture	41	8	37	37	37	8
Waveform2	40	13	21	22	40	24
Robot	24	13	24	24	24	24
Satimage	36	14	36	36	36	14
Ring	20	17	20	20	20	18

4.7.1 Evaluating Quality of Reduct Computed by FDM-FMFRS

Reduct computation in FDM-FMFRS is based on a discernibility matrix construction in the hyperbox space. Since fuzzy DM on IDS is an approximation of fuzzy DM on objects, theoretically, it results in an approximate reduct. Hence, naturally, it suffers from some information loss.

This section aims to assess the quality of approximate reduct obtained based on validation by computing the obtained gamma measure by reduct over the original decision system. The formulation of each algorithm uses its own FRS model to compute reduct. To avoid bias and to validate the relevance of reduct quality comparison, we needed to utilize a different FRS model so that the comparisons of gamma measures are with respect to a single FRS model. Similar to SAT measure, gamma measure is a widely employed dependency measure in FRS for accessing the quality of reduct. Hence, Gaussian kernel FRS (GKFRS) [31] is used for computation of gamma measure by reducts from the compared algorithms as none of these algorithms uses this particular approach (GKFRS) in their model.

Table 4.3 contains the resulting gamma value and reduct length by applying the proposed algorithm as well as the compared algorithms on the entire dataset. Also, Table 4.3 represents the gamma measure obtained from the unreduced decision system (mention as 'UNRED' in Table 4.3) to validate the relevance of resulted reducts through checking whether the obtained reduct is satisfying or reaching near to (UNRED) gamma measure or not.

Table 4.3 reports the gamma value (γ) for only eleven datasets out of seventeen benchmark datasets due to exceeding the memory limit while processing the GKFRS.

Analysis of Results

In Table 4.3, it is observed that FDM-FMFRS achieved the same gamma value as obtained by UNRED satisfying the required reduct property fully in Vehicle, Waveform2 and Ring datasets. In the remaining datasets, FDM-FMFRS achieved almost near to expected gamma measure w.r.t. entire dataset gamma value.

Overall, it can be seen that the approximate reduct from FDM-FMFRS is not resulting in any significant loss in the quality of reduct. Also, it can be observed that the size of reduct for FDM-FMFRS is much lesser than compared algorithms for all datasets. The compared algorithms have also achieved the relevant or approximate gamma measure in given datasets, but even that approximation is negligible, as in the case of FDM-FMFRS. Hence, empirically, we have established that FDM-FMFRS computed quality reduct with almost near gamma measure as UNRED and with a relatively shorter size reduct.

Section 4.7.2 explores the relevance of obtained approximate reduct of the FDM-FMFRS in achieving the construction of the classification learning model, which is the primary objective of the feature subset selection. Moreover, the comparative analysis with reduct length and computational time will be elaborated as part of Section 4.7.2 using tenfold cross-validation.

4.7.2 Relevance of the Proposed Approach in Construction of Classifiers

This section contains the comparative experiments conducted among algorithms for reduct computation, i.e., FDM-FMFRS, RMDPS [15], WRMDPS [15], FWARA [138] and PARA [72] algorithms. The relevance of reduct in inducing a classification model is studied through tenfold cross-validation (10-FCV) experiments. In each iteration, one fold is preserved for the testing data, and the remaining nine folds are used for training data. A reduct algorithm is applied to the training data. So, based on the reduct that is obtained, the classification model is constructed for comparison. The classification accuracy of the resulting model is evaluated based on the test data.

Two different classifier models are used, namely CART and kNN with default options, and for kNN experiments, k is taken as 3, and our proposed kNN-FMNN classifier (described in Chapter 3) is also employed for inducing classification model. To examine the relevance of reducts, we also construct the classification model with an unreduced dataset (mentioned as 'UNRED' in the given Tables) for comparison.

Table 4.4, Table 4.5 and Table 4.6 presents the results of the 10-FCV experiment for classification accuracies with CART, kNN, and kNN-FMNN respectively. Similarly, Table 4.7 and Table 4.8 illustrate the reduct length and computational time of the algorithms. Fig. 4.1, Fig. 4.2, Fig. 4.3, Fig. 4.4 and Fig. 4.5 depict the box-plot representation of results given in Table 4.4, Table 4.5, Table 4.6, Table 4.7 and Table 4.8 respectively.

The student's paired t-test with a significance level of 0.05 is performed in order to evaluate the statistical significance of the FDM-FMFRS algorithm with RMDPS, WRMDPS, FWARA, PARA and UNRED. Each column in Tables 4.4, 4.5, 4.6, 4.7 and 4.8 reports the results of the respective algorithm in the form of mean and standard deviation along with *p-value* except FDM-FMFRS column. FDM-FMFRS column contained only mean and standard deviation. The *p-value* index is the significant level between the respective algorithm and UNRED with FDM-FMFRS. If *p-value* > 0.05, then both approaches are no statistically significant difference and represented as a tie with the symbol of 'o'. For classification, if the *p-value* is less than equal to 0.05 and the result obtained by the respective algorithm is less than FDM-FMFRS, then the particular algorithm is statistically inferior to FDM-FMFRS and marked as a loss '-'. Otherwise, it is represented as a win '+'. The contrary is for reduct size and computational time analysis, which means if the *p-value* is less than 0.05, and the result obtained by the respective algorithm is less than FDM-FMFRS, then the particular algorithm is statistically significant than FDM-FMFRS and marked as a win '+'; otherwise, it is representing a loss '-'. For example, in classification Table 4.4 and computational time Table 4.7, the *p-value* column of PARA shows the '-' sign in Waveform2 dataset, which means that PARA is performing inferior to FDM-FMFRS in both classification and computational time.

The last three lines in each Table 4.4, 4.5, 4.6, 4.7 and 4.8 correspond to **Average (NOD)**, **CAverage**, and **Lose/Win/Tie**. It can be observed that the datasets over which an algorithm is executing vary from one to another. Hence, the average of individual mean values is reported in two forms. Average (NOD) corresponds to the average value obtained by an algorithm on datasets where it could be evaluated along with reporting the number of datasets (NOD) involved in brackets. CAverage value depicts the average of the individual mean obtained by restricting to only those datasets in which all algorithms could be evaluated. For the comparative analysis, CAverage plays an important role. The last line indicates the count of the number of statistically loss('-'), better('+'), and equivalent('o') for each algorithm in comparison with the proposed kNN-FMNN.

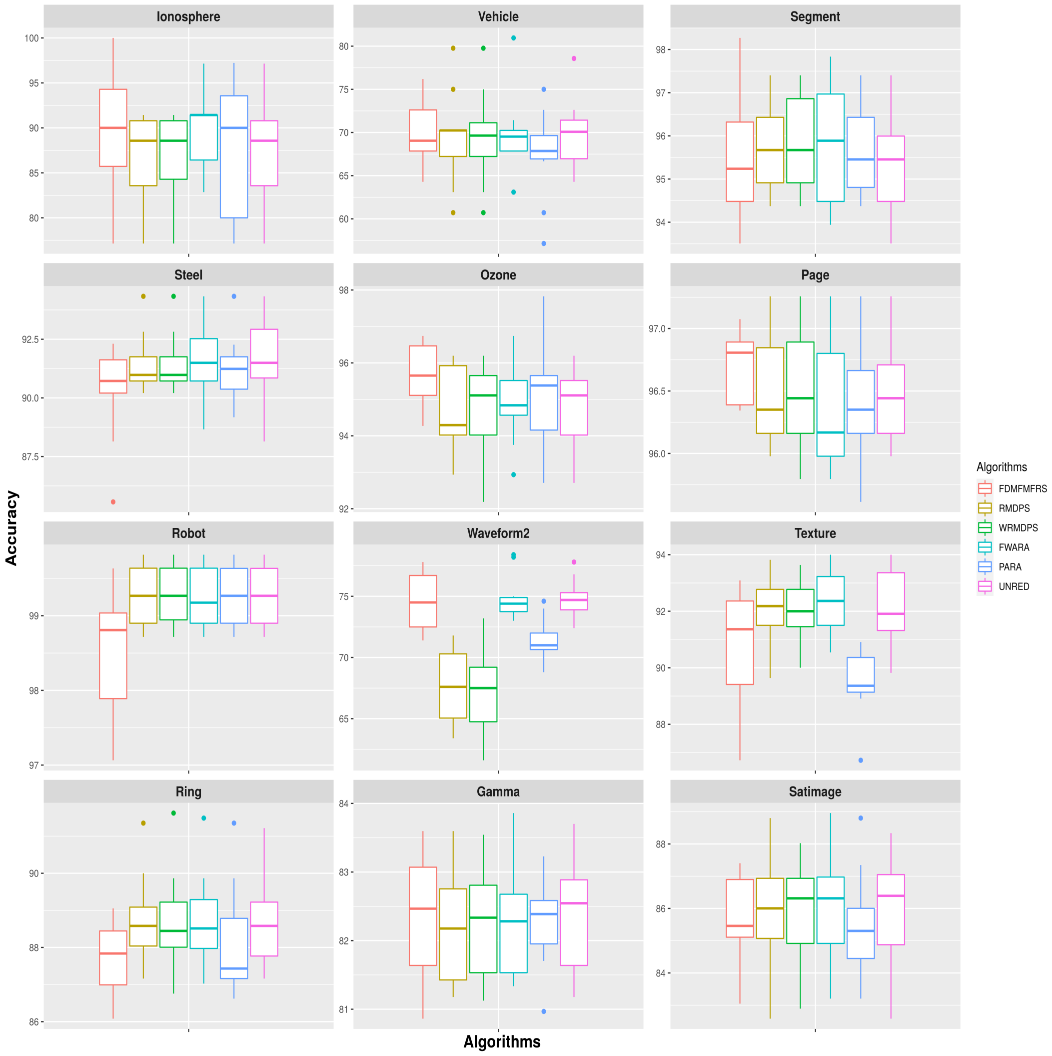
Note: The '*' sign in Tables 4.4, 4.5, 4.6, 4.7 and 4.8 shows the corresponding algorithm is intractable to a particular dataset to compute the reduct due to insufficient memory. And, '#' sign represents the scenario of non-termination of the code even after several hours of computation.

In Figures 4.1, 4.2, 4.3, 4.4 and 4.5, the range of Y-axis varies based on obtained results in each dataset. For large datasets, as the results are available only for FDM-FMFRS algorithm, Figures are respectively given in Figure (b) part.

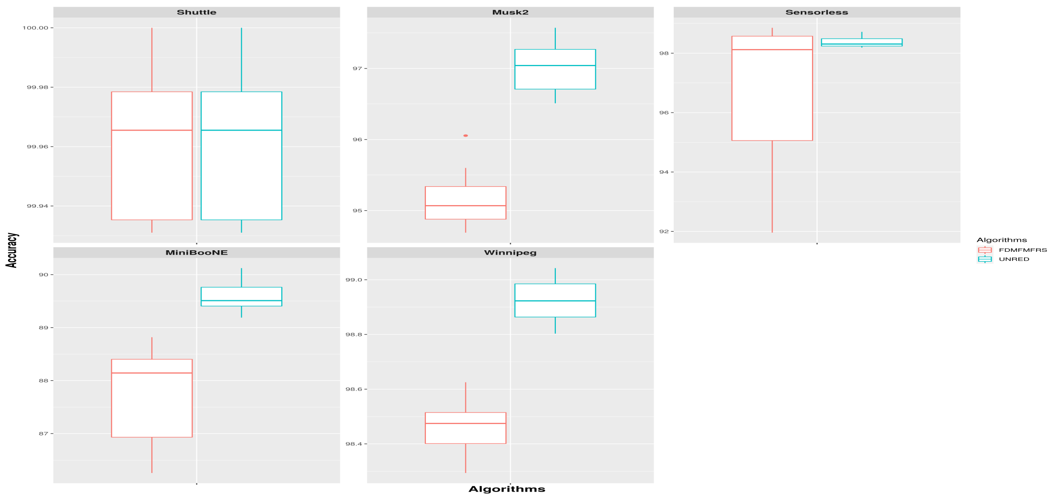
Table 4.4: Classification Accuracies Results (%) with CART in 10-FCV

Datasets	FDM-FMFRS	RMDPS		WRMDPS		FWARA		PARA		UNRED	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	88.87 ± 7.57	86.60 ± 5.06	0.44^o	86.89 ± 5.09	0.50^o	89.73 ± 4.72	0.76^o	87.15 ± 7.77	0.62^o	87.46 ± 5.59	0.64^o
Vehicle	70.17 ± 5.28	70.02 ± 5.71	0.95^o	70.03 ± 5.46	0.95^o	69.80 ± 4.50	0.87^o	67.37 ± 5.21	0.24^o	70.75 ± 4.56	0.80^o
Segment	95.45 ± 1.57	95.76 ± 1.00	0.61^o	95.80 ± 1.12	0.58^o	95.84 ± 1.42	0.57^o	95.62 ± 1.08	0.78^o	95.28 ± 1.27	0.79^o
Steel	90.26 ± 2.01	91.50 ± 1.25	0.12^o	91.50 ± 1.25	0.12^o	91.45 ± 1.64	0.17^o	91.24 ± 1.45	0.22^o	91.60 ± 1.88	0.14^o
Ozone	95.62 ± 0.90	94.65 ± 1.23	0.06^o	94.71 ± 1.40	0.10^o	94.97 ± 1.20	0.19^o	94.97 ± 1.50	0.25^o	94.65 ± 1.18	0.05^-
Page	96.71 ± 0.30	96.51 ± 0.50	0.29^o	96.49 ± 0.49	0.24^o	96.35 ± 0.52	0.07^o	96.43 ± 0.53	0.57^o	96.45 ± 0.42	0.14^o
Robot	98.63 ± 0.56	99.32 ± 0.25	0.00^+	99.32 ± 0.27	0.00^+	99.32 ± 0.26	0.00^+	99.32 ± 0.26	0.00^+	99.38 ± 0.25	0.00^+
Waveform2	74.66 ± 2.45	67.70 ± 2.97	0.00^-	67.10 ± 3.79	0.00^-	74.94 ± 1.86	0.78^o	71.44 ± 1.75	0.00^-	74.82 ± 1.58	0.86^o
Texture	90.82 ± 2.11	92.07 ± 1.22	0.12^o	92.02 ± 1.14	0.13^o	92.33 ± 1.17	0.06^o	89.47 ± 1.20	0.09^o	92.09 ± 1.37	0.13^o
Ring	87.70 ± 0.94	88.78 ± 1.18	0.04^+	88.69 ± 1.36	0.08^o	88.77 ± 1.27	0.05^+	88.12 ± 1.55	0.47^o	88.73 ± 1.27	0.05^+
Gamma	82.36 ± 0.93	82.20 ± 0.84	0.68^o	82.22 ± 0.82	0.73^o	82.27 ± 0.81	0.82^o	82.28 ± 0.66	0.81^o	82.34 ± 0.85	0.96^o
Satimage	85.75 ± 1.38	85.98 ± 1.78	0.75^o	86.00 ± 1.61	0.72^o	85.94 ± 1.80	0.80^o	85.54 ± 1.61	0.75^o	85.98 ± 1.70	0.74^o
Shuttle	99.96 ± 0.02	*		*		*		99.95 ± 0.02	0.27^o	99.96 ± 0.02	1.00^o
Musk2	95.16 ± 0.43	*		*		*		#		97.03 ± 0.36	0.00^+
Sensorless	97.54 ± 2.14	*		*		*		#		98.44 ± 0.22	0.20^o
MiniBooNE	88.75 ± 0.96	*		*		*		#		89.58 ± 0.28	0.01^+
Winnipeg	98.46 ± 0.10	*		*		*		#		98.92 ± 0.08	0.00^+
Average (NOD)	90.72 (17)	87.59 (12)		87.56 (12)		88.47 (12)		89.16 (13)		91.28 (17)	
CAverage$^{\$}$	88.08	87.59		87.56		88.47		87.41		88.29 (17)	
Lose/Win/Tie		1/2/9		1/1/10		0/2/10		1/1/11		1/5/11	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 12 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



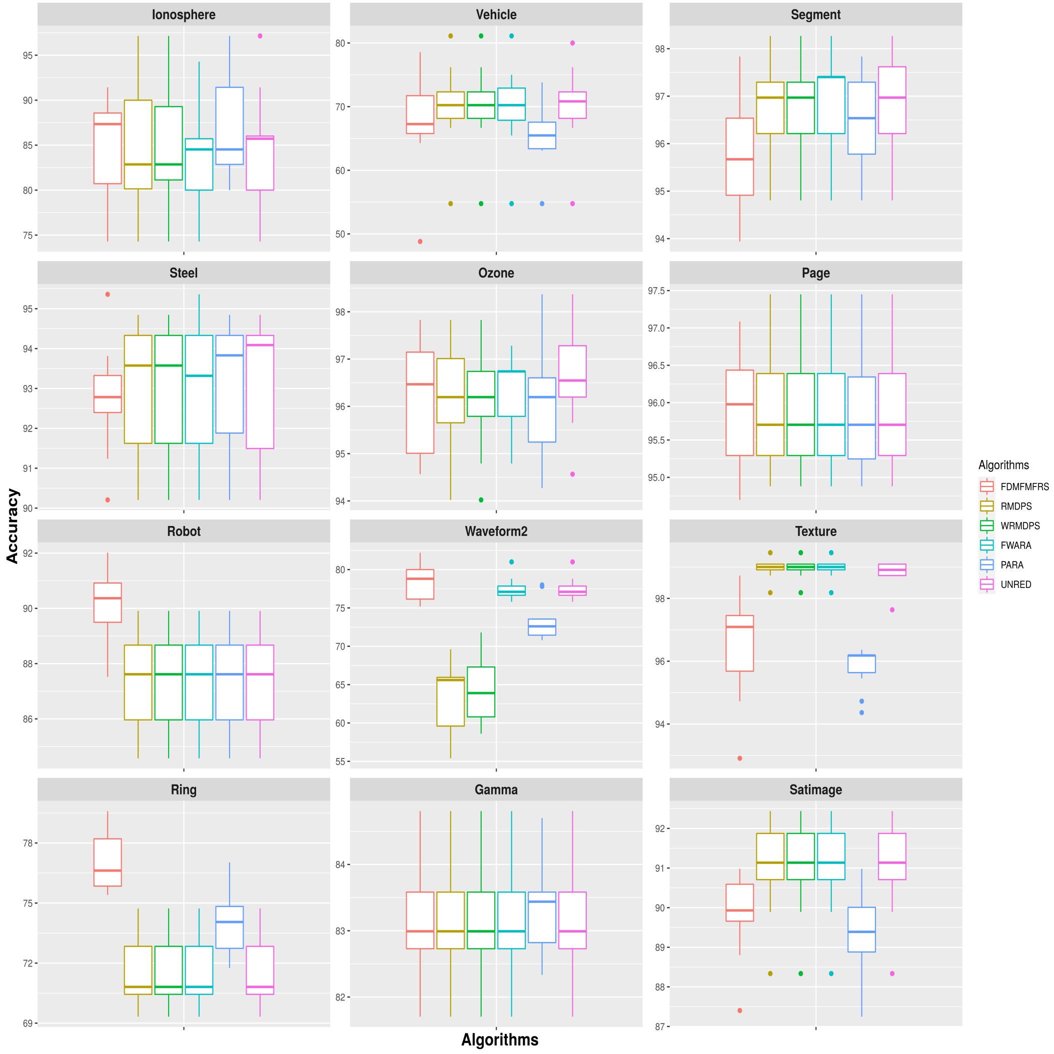
(b) Datasets having classification results by FDM-FMFRS and UNRED

Figure 4.1: Boxplot for Classification Accuracies Results with CART of Table 4.4

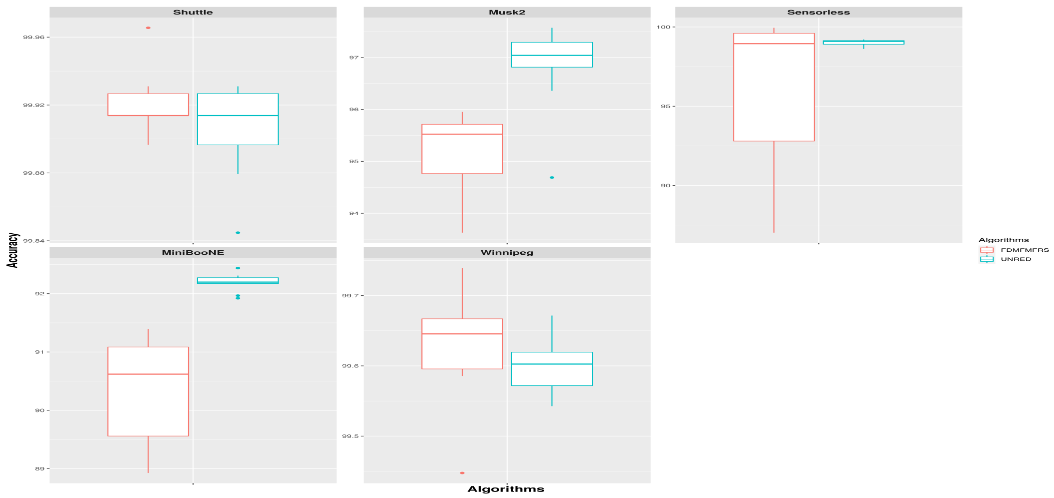
Table 4.5: Classification Accuracies Results (%) with kNN (k=3) in 10-FCV

Datasets	**FDM-FMFRS**	**RMDPS**		**WRMDPS**		**FWARA**		**PARA**		**UNRED**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	84.90 ± 5.40	84.91 ± 7.22	1.00^o	84.91 ± 7.09	1.00^o	83.19 ± 6.24	0.52^o	86.61 ± 5.84	0.50^o	84.04 ± 7.16	0.77^o
Vehicle	68.16 ± 4.54	69.60 ± 3.50	0.44^o	69.60 ± 3.50	0.44^o	69.35 ± 3.22	0.51^o	65.23 ± 4.85	0.18^o	69.60 ± 3.50	0.44^o
Segment	95.80 ± 1.28	96.75 ± 1.03	0.08^o	96.75 ± 1.03	0.08^o	96.88 ± 1.08	0.06^o	96.49 ± 1.04	0.20^o	96.80 ± 1.06	0.07^o
Steel	92.79 ± 1.40	93.04 ± 1.77	0.72^o	93.04 ± 1.77	0.72^o	93.10 ± 1.74	0.67^o	93.09 ± 1.59	0.65^o	93.15 ± 1.74	0.62^o
Ozone	96.16 ± 1.21	96.11 ± 1.17	0.92^o	96.22 ± 1.20	0.92^o	96.33 ± 0.87	0.73^o	96.05 ± 1.24	0.84^o	96.65 ± 1.11	0.36^o
Page	95.89 ± 0.82	95.85 ± 0.83	0.92^o	95.85 ± 0.83	0.92^o	95.85 ± 0.83	0.92^o	95.83 ± 0.82	0.91^o	95.85 ± 0.83	0.92^o
Robot	89.24 ± 0.98	87.37 ± 1.38	0.00^-	87.37 ± 1.38	0.00^-	87.37 ± 1.38	0.00^-	87.41 ± 1.85	0.01^-	87.37 ± 1.38	0.00^-
Waveform2	78.54 ± 2.53	63.38 ± 4.47	0.00^-	64.44 ± 4.81	0.00^-	77.50 ± 1.49	0.28^o	73.30 ± 2.60	0.00^-	77.50 ± 1.49	0.28^o
Texture	96.55 ± 1.73	98.98 ± 0.37	0.00^+	98.98 ± 0.37	0.00^+	98.98 ± 0.37	0.00^+	95.81 ± 0.72	0.22^o	98.80 ± 0.44	0.00^+
Ring	77.12 ± 1.60	71.62 ± 1.82	0.00^-	71.62 ± 1.82	0.00^-	71.62 ± 1.82	0.00^-	74.12 ± 1.74	0.00^-	71.62 ± 1.82	0.00^-
Gamma	83.13 ± 0.91	83.13 ± 0.91	1.00^o	83.13 ± 0.91	1.00^o	83.13 ± 0.91	1.00^o	83.13 ± 0.91	1.00^o	83.13 ± 0.91	1.00^o
Satimage	89.85 ± 1.09	91.05 ± 1.25	0.03^+	91.05 ± 1.25	0.03^+	91.05 ± 1.25	0.03^+	89.40 ± 1.09	0.36^o	91.05 ± 1.25	0.03^+
Shuttle	99.92 ± 0.02	*		*		*		99.90 ± 0.03	0.09^o	99.91 ± 0.03	0.21^o
Musk2	95.15 ± 0.82	*		*		*		#		96.83 ± 0.83	0.00^+
Sensorless	97.57 ± 3.61	*		*		*		#		99.03 ± 0.18	0.22^o
MiniBooNE	91.36 ± 0.92	*		*		*		#		92.19 ± 0.15	0.01^+
Winnipeg	99.63 ± 0.08	*		*		*		#		99.60 ± 0.04	0.29^o
Average (NOD)	90.17 (17)	85.98 (12)		86.08 (12)		87.02 (12)		87.89 (13)		90.39 (17)	
***CAverage*$^{\$}$**	87.34	85.98		86.08		87.02		86.37		87.13	
Lose/Win/Tie		3/2/7		3/2/7		2/2/8		3/0/10		2/4/11	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 12 datasets where all algorithms executed.
***** represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



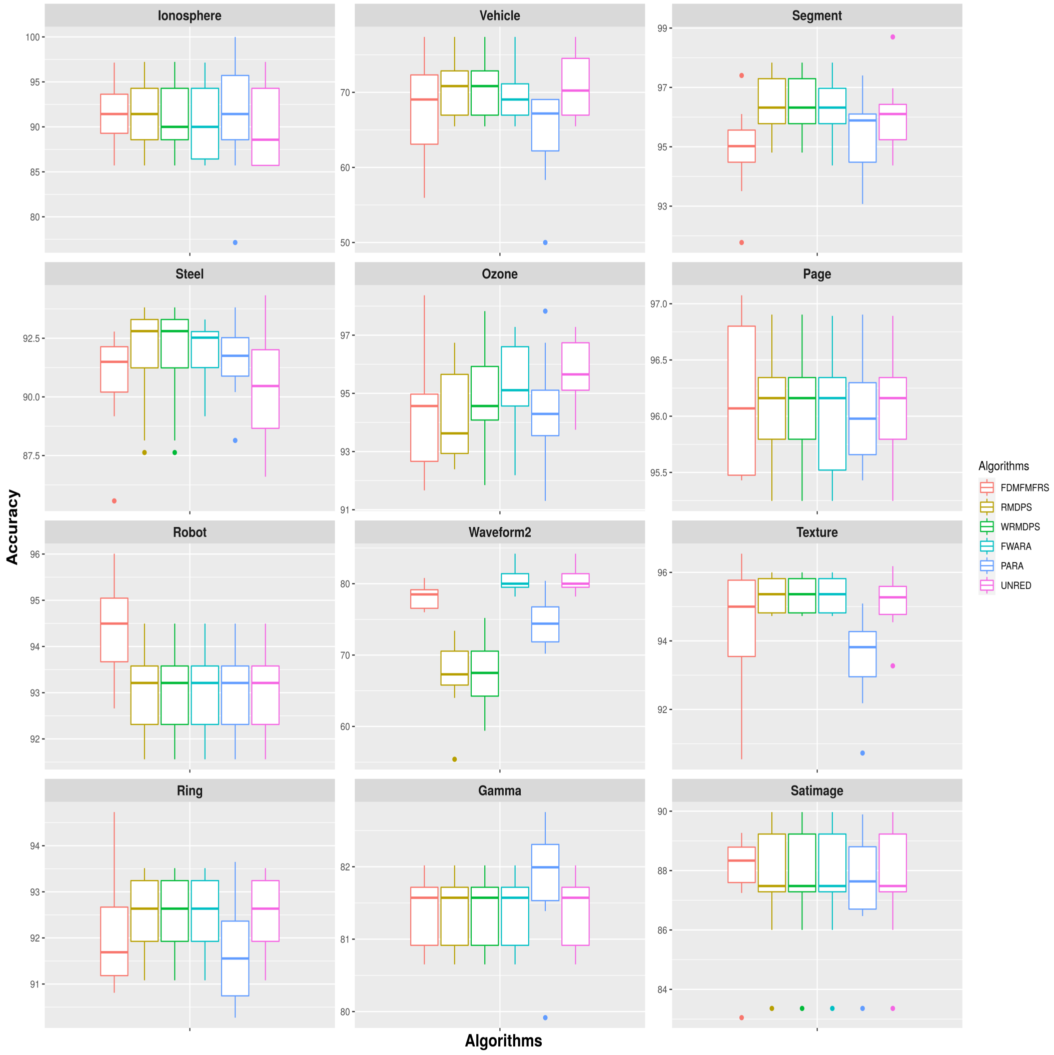
(b) Datasets having classification results by FDM-FMFRS and UNRED

Figure 4.2: Boxplot for Classification Accuracies Results with kNN of Table 4.5

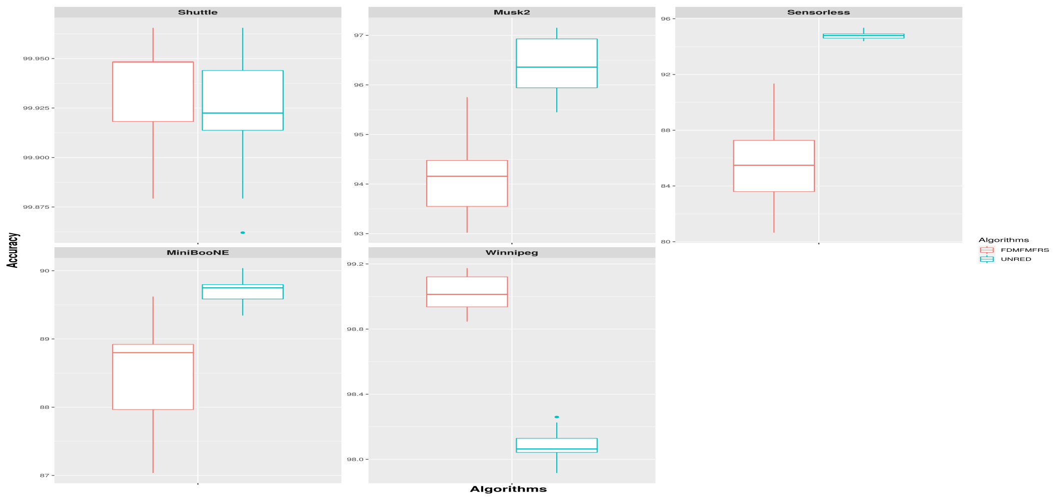
Table 4.6: Classification Accuracies Results (%) with kNN-FMNN in 10-FCV

Datasets	**FDM-FMFRS**	**RMDPS**		**WRMDPS**		**FWARA**		**PARA**		**UNRED**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	91.74 ± 3.67	91.44 ± 4.27	0.87^o	91.44 ± 4.05	0.86^o	90.59 ± 4.29	0.53^o	90.86 ± 6.57	0.71^o	90.29 ± 5.09	0.48^o
Vehicle	68.99 ± 4.19	68.08 ± 4.51	0.64^o	68.08 ± 4.51	0.64^o	69.87 ± 3.89	0.63^o	64.38 ± 6.32	0.07^o	68.56 ± 4.60	0.83^o
Segment	94.89 ± 1.51	96.45 ± 1.06	0.02^+	96.45 ± 1.06	0.02^+	96.32 ± 1.05	0.02^+	95.45 ± 1.27	0.38^o	96.02 ± 1.24	0.09^o
Steel	90.73 ± 2.14	91.81 ± 2.27	0.29^o	91.81 ± 2.27	0.29^o	91.81 ± 1.42	0.20^o	91.49 ± 1.57	0.37^o	90.37 ± 2.63	0.74^o
Ozone	94.17 ± 1.98	94.16 ± 1.62	0.99^o	94.92 ± 1.79	0.39^o	95.31 ± 1.52	0.17^o	94.42 ± 1.92	0.77^o	95.84 ± 1.15	0.03^+
Page	96.14 ± 0.67	96.11 ± 0.56	0.90^o	96.11 ± 0.56	0.90^o	96.03 ± 0.59	0.70^o	96.05 ± 0.49	0.73^o	96.09 ± 0.53	0.84^o
Robot	93.80 ± 1.28	93.24 ± 0.67	0.23^o	93.24 ± 0.67	0.23^o	93.24 ± 0.67	0.23^o	93.03 ± 0.96	0.14^o	93.24 ± 0.67	0.23^o
Waveform2	78.18 ± 1.66	67.04 ± 5.02	0.00^-	67.56 ± 4.79	0.00^-	80.58 ± 1.88	0.01^+	74.60 ± 3.42	0.00^-	80.58 ± 1.88	0.01^+
Texture	94.38 ± 1.93	95.36 ± 0.53	0.14^o	95.36 ± 0.53	0.14^o	95.36 ± 0.53	0.14^o	93.54 ± 1.34	0.27^o	95.11 ± 0.82	0.29^o
Ring	92.12 ± 1.31	92.47 ± 0.86	0.49^o	92.47 ± 0.86	0.49^o	92.47 ± 0.86	0.49^o	91.68 ± 1.16	0.43^o	92.47 ± 0.86	0.49^o
Gamma	81.38 ± 0.49	81.38 ± 0.49	1.00^o	81.38 ± 0.49	1.00^o	81.38 ± 0.49	1.00^o	81.38 ± 0.49	0.77^o	81.38 ± 0.49	1.00^o
Satimage	87.85 ± 1.82	87.61 ± 1.99	0.79^o	87.61 ± 1.99	0.79^o	87.61 ± 1.99	0.79^o	87.53 ± 1.86	0.70^o	87.61 ± 1.99	0.79^o
Shuttle	99.94 ± 0.03	*		*		*		99.91 ± 0.04	0.07^o	99.92 ± 0.03	0.33^o
Musk2	94.12 ± 0.76	*		*		*		#		96.36 ± 0.65	0.00^+
Sensorless	84.53 ± 2.22	*		*		*		#		94.85 ± 0.18	0.00^+
MiniBooNE	89.51 ± 0.83	*		*		*		#		89.71 ± 0.21	0.46^o
Winnipeg	99.02 ± 0.11	*		*		*		#		98.09 ± 0.10	0.00^-
Average (NOD)	90.34 (17)	87.92 (12)		88.02 (12)		89.21 (12)		89.22 (13)		91.31 (17)	
***CAverage*$^\$$**	88.69	87.92		88.02		89.21		87.86		88.96	
Lose/Win/Tie		11/0/1		1/1/10		0/2/10		1/0/12		1/3/12	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 12 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



(b) Datasets having classification results by FDM-FMFRS and UNRED

Figure 4.3: Boxplot for Classification Accuracies Results with kNN-FMNN of Table 4.6

Table 4.7: Computational Times Results (in seconds) in 10-FCV

Datasets	FDM-FMFRS	RMDPS		WRMDPS		FWARA		PARA	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	0.08 ± 0.01	0.20 ± 0.01	0.00^-	0.21 ± 0.00	0.00^-	0.70 ± 0.03	0.00^-	1.63 ± 0.09	0.00^-
Vehicle	0.08 ± 0.01	1.82 ± 0.04	0.00^-	1.96 ± 0.03	0.00^-	0.72 ± 0.03	0.00^-	2.59 ± 0.12	0.00^-
Segment	0.03 ± 0.00	15.96 ± 0.54	0.00^-	17.29 ± 0.55	0.00^-	1.60 ± 0.04	0.00^-	14.38 ± 0.49	0.00^-
Steel	0.61 ± 0.04	2.17 ± 0.11	0.00^-	2.35 ± 0.14	0.00^-	2.88 ± 0.18	0.00^-	17.31 ± 0.94	0.00^-
Ozone	1.24 ± 0.04	0.90 ± 0.04	0.00^-	0.97 ± 0.05	0.00^-	21.12 ± 1.57	0.00^-	33.68 ± 2.71	0.00^-
Page	0.07 ± 0.00	17.47 ± 0.26	0.00^-	18.58 ± 0.29	0.00^-	3.27 ± 0.03	0.00^-	13.69 ± 1.15	0.00^-
Robot	4.85 ± 0.30	73.89 ± 1.27	0.00^-	76.78 ± 1.14	0.00^-	35.95 ± 0.44	0.00^-	765.90 ± 8.04	0.00^-
Waveform2	18.86 ± 0.48	66.76 ± 0.36	0.00^-	70.59 ± 0.84	0.00^-	188.27 ± 2.94	0.00^-	1279.6 ± 0.02	0.00^-
Texture	0.08 ± 0.02	105.28 ± 0.65	0.00^-	110.88 ± 0.72	0.00^-	50.34 ± 0.98	0.00^-	155.37 ± 8.53	0.00^-
Ring	5.75 ± 0.05	98.18 ± 0.67	0.00^-	103.34 ± 0.85	0.00^-	27.87 ± 0.49	0.00^-	355.58 ± 21.57	0.00^-
Gamma	1.77 ± 0.04	549.37 ± 17.86	0.00^-	569.81 ± 10.67	0.00^-	46.83 ± 0.28	0.00^-	255.26 ± 7.46	0.00^-
Satimage	0.85 ± 0.04	131.86 ± 1.18	0.00^-	138.36 ± 1.64	0.00^-	82.40 ± 2.13	0.00^-	307.65 ± 12.37	0.00^-
Shuttle	0.44 ± 0.03	*		*		*		1243.4 ± 0.06	0.00^-
Musk2	26.22 ± 1.38	*		*		*		#	
Sensorless	0.37 ± 0.03	*		*		*		#	
MiniBooNE	287.48 ± 11.21	*		*		*		#	
Winnipeg	1527.67 ± 20.59	*		*		*		#	
Average (NOD)	110.37 (17)	88.65 (12)		92.59 (12)		38.49 (12)		321.44 (13)	
CAverage$^{\$}$	2.85	88.65		92.59		38.49		266.88	
Lose/Win/Tie		12/0/0		12/0/0		12/0/0		13/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).

$: Average mean value over 12 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.

Figure 4.4: Boxplot for Computational Time Results of Table 4.7

Table 4.8: Reduct Length Results in 10-FCV

Datasets	FDM-FMFRS	RMDPS		WRMDPS		FWARA		PARA	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	6.60 ± 0.70	26.70 ± 1.06	0.00^-	27.60 ± 1.35	0.00^-	30.80 ± 0.79	0.00^-	16.70 ± 0.82	0.00^-
Vehicle	13.40 ± 1.58	17.80 ± 0.42	0.00^-	17.80 ± 0.42	0.00^-	16.60 ± 0.52	0.00^-	8.60 ± 0.51	0.00^+
Segment	8.10 ± 0.88	15.00 ± 0.00	0.00^-	15.00 ± 0.00	0.00^-	14.10 ± 0.32	0.00^-	9.20 ± 0.42	0.00^-
Steel	12.80 ± 1.62	20.00 ± 0.94	0.00^-	20.00 ± 0.94	0.00^-	17.20 ± 0.63	0.00^-	14.90 ± 0.31	0.00^-
Ozone	9.60 ± 0.84	37.40 ± 1.07	0.00^-	39.40 ± 1.78	0.00^-	50.90 ± 1.91	0.00^-	28.10 ± 0.87	0.00^-
Page	7.50 ± 1.27	9.90 ± 0.32	0.00^-	9.90 ± 0.32	0.00^-	9.90 ± 0.32	0.00^-	8.80 ± 0.42	0.00^-
Robot	12.80 ± 0.92	24.00 ± 0.00	0.00^-	24.00 ± 0.00	0.00^-	24.00 ± 0.00	0.00^-	24.00 ± 0.00	0.00^-
Waveform2	13.00 ± 0.00	21.20 ± 0.42	0.00^-	21.60 ± 0.70	0.00^-	40.00 ± 0.00	0.00^-	23.90 ± 0.56	0.00^-
Texture	8.60 ± 1.26	37.00 ± 0.00	0.00^-	37.00 ± 0.00	0.00^-	37.00 ± 0.00	0.00^-	6.90 ± 0.56	0.00^+
Ring	16.00 ± 0.00	20.00 ± 0.00	0.00^-	20.00 ± 0.00	0.00^-	20.00 ± 0.00	0.00^-	17.90 ± 0.31	0.00^-
Gamma	10.00 ± 0.00	10.00 ± 0.00	1.00^o	10.00 ± 0.00	1.00^o	10.00 ± 0.00	1.00^o	9.00 ± 0.00	0.00^+
Satimage	15.20 ± 1.69	36.00 ± 0.00	0.00^-	36.00 ± 0.00	0.00^-	36.00 ± 0.00	0.00^-	12.20 ± 0.91	0.00^+
Shuttle	4.90 ± 0.32	*		*		*		6.00 ± 0.00	0.00^-
Musk2	13.50 ± 0.53	*		*		*		#	
Sensorless	9.50 ± 1.18	*		*		*		#	
MiniBooNE	22.80 ± 0.79	*		*		*		#	
Winnipeg	18.70 ± 0.67	*		*		*		#	
Average (NOD)	11.94 (17)	22.54 (12)		22.80 (12)		25.11 (12)		14.58 (13)	
CAverage$^{\$}$	11.13	22.54		22.80		25.11		15.07	
Lose/Win/Tie		11/0/1		11/0/1		11/0/1		9/4/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).

$: Average mean value over 12 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.

4.7.3 Analysis of Results

Classification Results

Table 4.4, Table 4.5 and Table 4.6 show the classification results of CART, kNN and kNN-FMNN classifiers. In all classifiers, the CAverage value of the proposed algorithm FDM-FMFRS is higher than compared algorithms and very near to UNRED.

In Table 4.4, considering the overall 66 accuracy results across all the compared algorithms and UNRED in CART classifier, the cumulative lose/win/tie results are 4/11/51. Hence in the majority of results (51), the proposed algorithm FDM-FMFRS performed statistically similar to compared algorithms and UNRED. Also, it is observed that wherever FDM-FMFRS performed a little inferior to compared algorithms and UNRED (i.e., 11 results), the differences in average mean are very small. In the remaining 4 results, the proposed algorithm FDM-FMFRS performed significantly better than the compared algorithms, and here also, it is observed that the difference in mean value is small.

Similarly, in other kNN and kNN-FMNN classifiers, as given in Table 4.5 and Table 4.6, majorly all algorithms performed statistically similar to each other. The cumulative lose/win/tie results in kNN classifier is 13/10/43 and in kNN-FMNN is 4/7/55. The further observation analysis details are given below.

FDM-FMFRS achieved statistically better than RMDPS, WRMDPS and PARA algorithms in Waveform2 dataset in all classifiers, as shown in Fig. 4.1, 4.2 and 4.3. In Musk2 dataset, FDM-FMFRS performed statistically inferior to UNRED.

Based on results given in Table 4.4 and Fig. 4.1, for Ring, MiniBooNE and Winnipeg datasets, FDM-FMFRS incurred statistically inferior to UNRED, but the difference in average classification accuracies for both algorithms is insignificant on datasets, for example, In Winnipeg, FDM-FMFRS is 98.46 and UNRED is 98.92. Similarly, in Robot datasets, FDM-FMFRS obtained statistically inferior results than compared algorithms (including UNRED), but the difference in their results is almost quite low. Moreover, FDM-FMFRS resulted statistically better than UNRED in Ozone dataset.

The following observations are made for the results given in Table 4.5 and Fig. 4.2 using kNN classifier. FDM-FMFRS achieved statistically better results than compared algorithms (including UNRED) in Robot and Ring datasets. In contrast, FDM-FMFRS is statistically inferior to compared algorithms (except PARA) and UNRED in Texture and Satimage datasets, but the difference in their average classification accuracies are very less on these datasets.

Similarly, in kNN-FMNN classifier, based on Table 4.6 and Fig. 4.3, FDM-FMFRS obtained statistically better than UNRED in Winnipeg dataset. However, FDM-FMFRS performed statistically inferior to UNRED in Sensorless dataset with a high difference in their

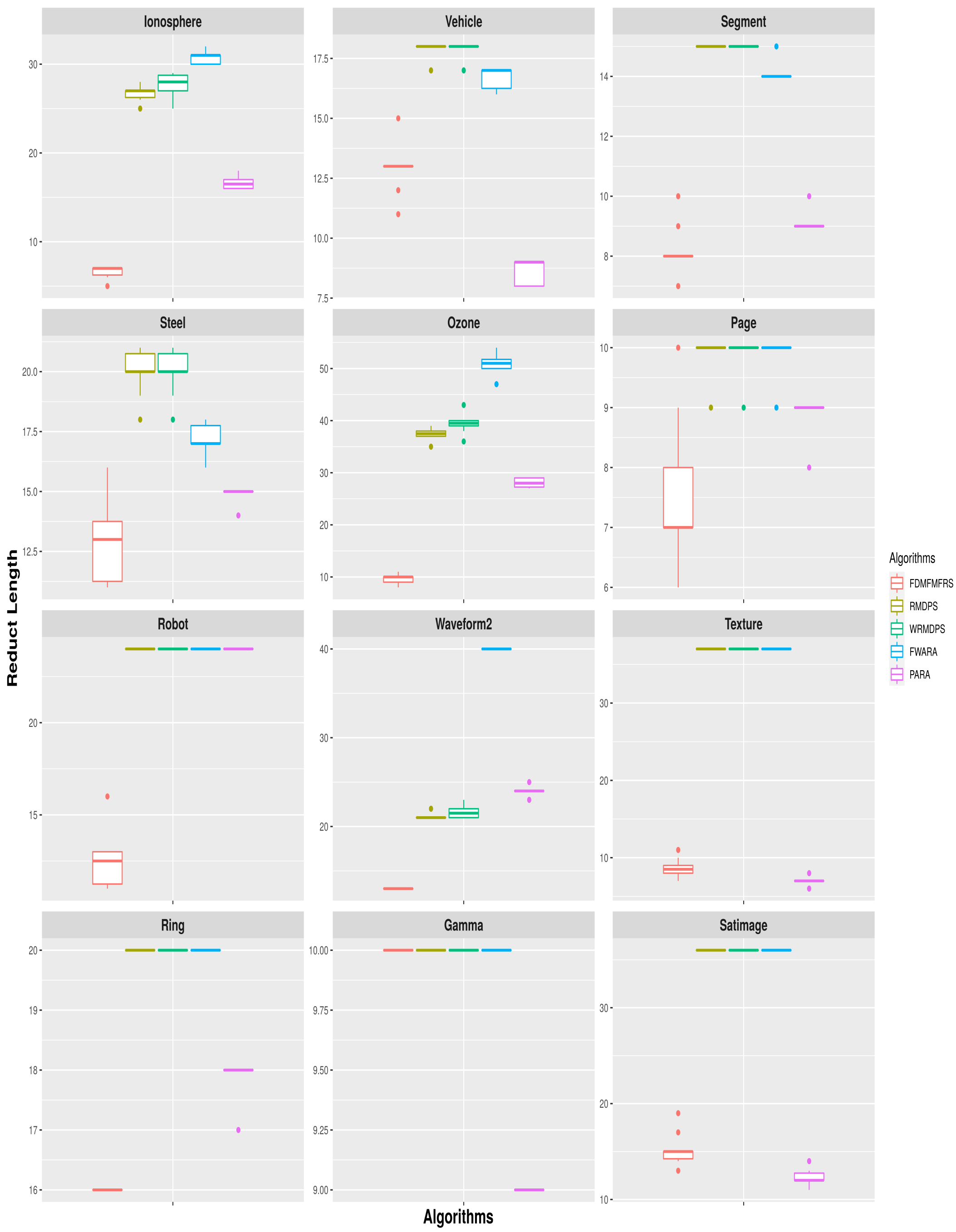


Figure 4.5: Boxplot for Reduct Length Results of Table 4.8

average classification accuracy.

In Sensorless dataset, FDM-FMFRS incurred much similar classification results in CART and kNN classifiers; however, it suffered a little bit in kNN-FMNN classifier. But as we can observe, it obtained a much more significant reduct with a substantial reduction in the size of the actual attributes from 58 to 9.

In Musk2 dataset, FDM-FMFRS performed statistically inferior to compared algorithms in all classifiers, but there is not much difference in mean value. And the decrease in classification accuracy might be due to the reduction of attributes from 166 to 22.

In MiniBooNE and Winnipeg datasets, FDM-FMFRS incurred less significantly than UNRED in CART and kNN classifiers, but their difference in classification accuracies is very minor, for example, in kNN classifier, FDM-FMFRS got 91.36% accuracy where UNRED got 92.19%. In sensorless and musk2 datasets, FDM-FMFRS performed less significant than UNRED in given classifiers.

Hence, in most of the datasets, FDM-FMFRS has obtained similar or better classification accuracy than compared algorithms, and in those results where FDM-FMFRS has performed inferior, their mean accuracy is very near. Hence, on the whole, one can conclude that the approximate reduct through FDM-FMFRS preserved the quality of reduct in inducing a good classification model. We can see that the average value of the individual mean of classification accuracy of the FDM-FMFRS algorithm on overall datasets is quite near the average value results in UNRED, which shows effectiveness in classification performance. Also, It is further observed that RMDPS, WRMDPS, FWARA and PARA algorithms could not obtain reduct in Shuttle, Musk2, Sensorless, MinibooNE and Winnepeg datasets due to memory overflow (Sign '*') or non-termination even after 24 hours (Sign '#') at given system configuration where FDM-FMFRS can obtain reduct in few seconds. Eventually, it can be seen that the idea of computing the approximate reduct by FDM-FMFRS is satisfactory and effective.

Computational Time Results

In terms of computational times, as shown given in Table 4.7 and Fig. 4.4, FDM-FMFRS incurred significantly less computational time than compared algorithms for all datasets and evidently seen that the cumulative lose/win/tie results of compared algorithms are 49/0/0. Also, the proposed method FDM-FMFRS obtained the least CAverage value (2.85 seconds), which is significantly lesser than compared algorithms with CAverage in the range of 38 to 266 seconds.

The average mean value of FDM-FMFRS on 17 datasets is 110.37 seconds. But, none of the compared algorithms could scale to all 17 datasets. This significantly established that

FDM-FMFRS is computationally scalable than all compared algorithms. Even the resulting standard deviation of computation time presented very little variation, thus showing that the methodology is reliable compared to others.

These substantial reductions of computational time of FDM-FMFRS are due to the dealing with hyperboxes constructed by FMNN model where $|HBS| << |U|$. In order to understand the significance of space complexity reduction in fuzzy DM size, we compared $|U|^2$ with an average of $|HBS|^2$ in 10-FCV. The results are shown in Table 4.9. Table 4.9 provides the average and standard deviation of the obtained number of hyperboxes (NOH), the average value of $|HBS|^2$ obtained in fuzzy DM size (PHDS) and the percentage of reduction (POR) of HBS based fuzzy DM size over U based fuzzy DM size. Based on Table 4.9, the percentage of reduction (POR) of $|HBS|^2$ over $|U|^2$ got in the range of 78-99% across the given datasets. One can say that a pair of hyperboxes comparison in hyperbox based-fuzzy DM absorbs many pairs of objects comparison in object based-fuzzy DM (i.e., $|HBS| << |U|$). Owing to this significant reduction, FDM-FMFRS could be applied on such datasets where compared algorithms (RMDPS, WRMDPS, FWARA and PARA) would not execute, as the required memory space for these datasets is not available in the given system considered.

Traditional or, Scalable FRS time complexity is $O(|U|^2|C^n|^2)$ which hinders the applicability to large decision systems, whereas FDM-FMFRS achieves $O(|HBS|^2|C^n|^2)$ against $O(|U|^2|C^n|^2)$ which enhance scalability. FDM-FMFRS outperformed and saved more than 90–99% of the average computational time than other compared algorithms. Thus, the speed-up computation and performance demonstrate the potential of FDM-FMFRS algorithm and its suitability for larger datasets.

Hence, working through granular computing and performing feature subset selection at the hyperbox level has resulted in obtaining a quality reduct with scalability.

Reduct length Results

The results given in Table 4.8 and Fig. 4.5 established that FDM-FMFRS obtained reduct with statistically lesser size than RMDPS, WRMDPS and FWARA for all datasets except Gamma dataset and evidently seen that the cumulative lose/win/tie results of compared algorithms are 42/4/3. In Gamma dataset, all algorithms including the proposed work FDM-FMFRS obtained entire attributes as reduct. FDM-FMFRS got a statistically larger reduct size than PARA with Vehicle, Texture, Gamma and Satimage datasets, but the quality of reduct from FDM-FMFRS in terms of average classification accuracies statistically is not compromised.

The experimental results established that the applicability of the FRS reduct algorithm is enhanced strongly with FMNN preprocessing. The proposed approach FDM-FMFRS exhibits

Table 4.9: Reduction of Datasets with FMNN as a Preprocessor

Datasets	EDS	NOH	PHDS	POR
	$\|U\|^2$	Mean ± Std		
Ionosphere	351 × 351	79.60 ± 2.80	80 × 80	77.20%
Vehicle	846 × 846	71.40 ± 5.21	71 × 71	91.60%
Segment	2310 × 2310	33.10 ± 1.66	33 × 33	98.57%
Steel	1941 × 1941	188.10 ± 6.97	188 × 188	90.31%
Ozone	1848 × 1848	279.00 ± 2.94	279 × 279	84.90%
Page	5472 × 5472	24.30 ± 1.25	24 × 24	99.56%
Robot	5456 × 5456	578.70 ± 11.22	578 × 578	89.40%
Waveform2	5000 × 5000	963.20 ± 3.33	963 × 963	80.74%
Texture	5500 × 5500	44.50 ± 2.17	45 × 45	99.18%
Gamma	19020 × 19020	302.20 ± 8.16	302 × 302	98.41%
Satimage	6435 × 6435	218.67 ± 218.67	219 × 219	96.59%
Ring	7400 × 7400	630.70 ± 3.27	631 × 631	91.47%
Musk2	6598 × 6598	751.40 ± 10.44	751 × 751	88.61%
Shuttle	57999 × 57999	14.30 ± 0.67	14 × 14	99.97%
Sensorless	58509 × 58509	26.10 ± 1.52	26 × 26	99.95%
MiniBooNE	129596 × 129596	1938.20 ± 13.35	1938 × 1938	98.50%
Winnipeg	325834 × 325834	3591.80 ± 22.65	3591 × 3591	98.89%

Notes: EDS: Estimated Fuzzy DM sizes, NOH: Number of hyperboxes, PHDS: Proportional Fuzzy DM sizes, POR: Percentage of reduction.

enhanced scalability on large datasets and induce better or similar classification performance with relevant reduct.

4.8 Summary

We proposed FDM-FMFRS as a hybridization of FMNN with FRS for reduct computation, intending to increase scalability on benchmark datasets. Here, we replaced DM construction in object space with hyperbox space which is obtained through FMNN. A hyperbox based fuzzy DM construction approximated traditional DM, so that the computed reduct is also an approximate reduct. The extensive experimental study was done with state-of-the-art FRS approaches on several benchmark datasets to establish the relevance of FDM-FMFRS reduct. And results demonstrated that FDM-FMFRS achieved significant computational gains over existing state-of-the-art FRS approaches while achieving similar or better classification accuracies. Also, FDM-FMFRS could scale to such large datasets where existing FRS algorithms are unable to compute due to space constraints.

Chapter 5

Variant of FDM-FMFRS for Feature Subset Selection

This chapter explores an extension of FDM-FMFRS approach discussed in chapter 4. In chapter 4, FDM-FMFRS approach has enhanced the scalability FRS based attribute reduction to large datasets due to the construction of fuzzy DM in hyperbox space instead of object space. This chapter investigates a scenario emerging through modification and adaptation in FDM-FMFRS that can enhance further scalability in hyperbox space. We propose an alternative approach even though FDM-FMFRS is complete and sufficient. This design can tune FDM-FMFRS approach being applicable to much larger size datasets.

The rest of the chapter is designed as follows: Section 5.1 present the brief introduction. Section 5.2 introduces the motivation of the proposed algorithm. Section 5.3 describes the functioning of the proposed algorithm CDM-FMFRS. Section 5.4 describes the complexity analysis of proposed algorithm CDM-FMFRS. Section 5.5 reports a series of experiments and comparative analysis of CDM-FMFRS with FDM-FMFRS and state-of-the-art approaches.

5.1 Motivation

In the previous chapter 4, we adopt FMNN learning model as a preprocessor to work on granular based computing for FRS reduct computation (FDM-FMFRS). FDM-FMFRS approach indeed enhanced the scalability in large decision systems due to the construction of fuzzy DM in hyperbox space instead of object space. Also, FDM-FMFRS approach significantly decreases the computation and space complexity on several benchmark datasets in a given memory constraint.

In this chapter, we are improvising the performance of FDM-FMFRS in terms of scalability. We further increase the scalability of reduct computation in hyperbox space in FDM-

FMFRS approach. Hence, our work formulates a way to reduce space utilization of fuzzy DM in paving the way to increased scalability. An approach is proposed by adopting the crisp DM construction over fuzzy DM construction in hyperbox space that can increase the scalability of datasets on the given memory constraints. The formation of crisp DM naturally incurs information loss. So, we also enriched crisp DM with a defined tolerance parameter to facilitate the perseverance of potential attributes in crisp DM entries.

5.2 Space Utilization of Fuzzy DM vs Crisp DM

The enhanced scalability in FDM-FMFRS is because of the construction of fuzzy DM in hyperbox space. As it is demonstrated in the experiment section 4.7.2 for a very large dataset, the cardinality of hyperbox space itself grows to such large numbers such that the construction of fuzzy DM in hyperbox space itself is not permissible. So, our work aims at overcoming this limitation to further increase the scalability of hyperbox space-based FRS reduct computation.

The sections 4.2.1.1 and 4.2.2.1 introduce the concept of crisp DM and fuzzy DM. Theoretically, the space complexity of both approaches in hyperbox space is $O(|HBS|^2|C^n|)$. An entry of crisp DM is a subset of C^n, whereas an entry of fuzzy DM is a real-valued array of size C^n. Adapting the characteristic function for the representation of a subset of C^n and using bitset representation for same, the entry of crisp DM requires $|C^n|$ bits. Assume that a real-valued numbered is represented in the computer using 'k' bytes. Then it follows that the space utilization of crisp DM is $\frac{1}{8\times k}$ of the space utilization of fuzzy DM. Hence, the reduction in space utilization in crisp DM construction is highly significant.

However, the construction of crisp DM for a numerical decision system involves information loss. In the literature, it is arrived at by discretization of the numerical decision system or application of threshold fuzzy discernibility value [45]. Either way, a lot of information in fuzzy DM is lost in the conversion to crisp DM. So it becomes imperative to arrive at a crisp DM formulation for availing of space reduction using a methodology aiming at lessening the information loss.

Hence, with the objective of further increasing the scalability of FDM-FMFRS, we proposed a novel crisp DM formulation instead of fuzzy DM for FRS reduct computation. The adaptation of crisp DM for FRS reduct computation is motivated from works [10, 15, 112, 119] that significantly reduces the space complexity.

5.3 Proposed CDM-FMFRS Reduct Algorithm

In this section, we propose an approach CDM-FMFRS (CDM: Crisp discernibility matrix, FM: Fuzzy min-max neural network, FRS: Fuzzy rough set) to increase the scalability of FDM-FMFRS in hyperbox space. This paper also aims to compute an approximate reduct efficiently with significant gains in space and time complexity. The proposed work (CDM-FMFRS) is summarized as follows:

1. Creation of interval-valued decision system (IDS) from FMNN preprocessing.
2. Crisp Discernibility matrix construction based on interval-valued decision system.
3. Compute an approximate reduct computation based on crisp discernibility matrix.

Moreover, we incorporate the following features in crisp DM formation with the objective of minimizing the inevitable information loss and preserving potential attributes as part of discernibility matrix entries. Furthermore, we extend the overlapping criteria amidst hyperboxes with three additional rules in achieving crisp DM formulation and also enrich with a defined tolerance parameter to facilitate the perseverance of potential attributes in crisp discernibility relation through hyperboxes.

5.3.1 Creation of Interval-Valued Decision System from FMNN

In the proposed CDM-FMFRS, the construction of IDS based on fuzzy hyperboxes is done as per the procedure given and described in the Section 4.5.1.

5.3.2 FMNN Preprocessor based crisp Discernibility Matrix

Here, we provide the procedure for formulation of crisp DM on IDS. This is based on discernibility between two hyperboxes. Each entry $M(H_i, H_j)$ in crisp DM is obtained from a pair of hyperboxes H_i and H_j of different classes. Each clause contains a set of attributes that have non-overlapping or allowable proportions (user defined parameter θ_1) of overlapping intervals between hyperboxes H_i and H_j. Each clause $M(H_i, H_j)$ is defined in Eqn. (5.1).

$$\begin{aligned} M(H_i, H_j) = \big\{a \mid a \in C \wedge \ OverlapInDim(H_i, H_j, a) == False \vee \\ (OverlapInDim(H_i, H_j, a) == True \wedge propoverlap < \theta_1)\big\} \end{aligned} \tag{5.1}$$

The expression $OverlapInDim(H_i, H_j, a)$ performs the overlap test between hyperboxes H_i and H_j along 'a' dimension. Simpson [102] introduces the four conditions to check the overlapping along a particular dimension in FMNN model. In [112], authors extend the conditions for overlapping cases when min point and max point coincide at the considered dimension. We have further introduced three more conditions for accommodating overlapping in the case at least one of the hyperbox is a point hyperbox. The following are the eleven conditions over which overlapping status is determined and can be considered as a complete set of conditions for checking overlap. The following 11 cases contain the possible way of overlapping a dimension between hyperboxes to become true ($OverlapInDim(H_i, H_j, a) == True$).

$$
\begin{aligned}
case\ 1 &: v_a^{H_i} == w_a^{H_i}\ and\ v_a^{H_j} == w_a^{H_j}\ and\ v_a^{H_i} == v_a^{H_j} \\
case\ 2 &: v_a^{H_i} == w_a^{H_i}\ and\ v_a^{H_j} \neq w_a^{H_j} \\
&\quad if\ (v_a^{H_j} \leq v_a^{H_i}\ and\ v_a^{H_i} \leq w_a^{H_j}) \\
case\ 3 &: v_a^{H_i} \neq w_a^{H_i}\ and\ v_a^{H_j} == w_a^{H_j} \\
&\quad if\ (v_a^{H_i} \leq v_a^{H_j}\ and\ v_a^{H_j} \leq w_a^{H_i}) \\
case\ 4 &: v_a^{H_i} < v_a^{Hj} < w_a^{H_i} < w_a^{Hj} \\
&\quad o_p = w_a^{H_i} - v_a^{Hj} \\
case\ 5 &: v_a^{H_j} < v_a^{H_i} < w_a^{H_j} < w_a^{H_i} \\
&\quad o_p = w_a^{H_j} - v_a^{H_i} \\
case\ 6 &: v_a^{H_i} < v_a^{Hj} < w_a^{Hj} < w_a^{H_i} \\
&\quad o_p = w_a^{H_j} - v_a^{H_j} \\
case\ 7 &: v_a^{H_j} < v_a^{H_i} < w_a^{H_i} < w_a^{H_j} \\
&\quad o_p = w_a^{H_i} - v_a^{H_i} \\
case\ 8 &: v_a^{H_i} = v_a^{Hj} < w_a^{H_i} < w_a^{H_j} \\
&\quad o_p = w_a^{H_i} - v_a^{H_j} \\
case\ 9 &: v_a^{H_j} < v_a^{H_i} < w_a^{Hj} = w_a^{H_i} \\
&\quad o_p = w_a^{H_j} - v_a^{H_i} \\
case\ 10 &: v_a^{H_i} = v_a^{Hj} < w_a^{Hj} < w_a^{H_i} \\
&\quad o_p = w_a^{Hj} - v_a^{Hj} \\
case\ 11 &: v_a^{H_j} < v_a^{H_i} < w_a^{H_i} = w_a^{H_j} \\
&\quad o_p = w_a^{H_i} - v_a^{H_i}
\end{aligned}
\tag{5.2}
$$

Cases from first to third correspond to newly introduced overlapping conditions for point hyperboxes, cases 4th to 7th correspond to overlapping conditions in FMNN [102], and the remaining cases are the additional conditions introduced in EFMNN [63]. In each overlap step, we are adding the partial overlapping check (Eqn. (5.3)) based on the proportion between hyperboxes.

The importance of considering partial overlapping steps between hyperboxes lessens the imposition of rigid rules such as sufficient separability between the hyperboxes along the chosen dimensions that can result in significant information loss and possibly a sparse DM. Even if two hyperboxes have a slight overlap in a dimension, then there is a sufficient chance that the attribute is discerning most of the objects of one hyperbox from that of another hyperbox. Preserving such discernible attributes in the crisp DM formation is very important in minimizing the information loss in crisp DM formation. Hence, the following properties are arrived at for deciding when an attribute becomes a discernible attribute.

1. An attribute is considered as discerning, if it is a non-overlapping dimension.
2. An attribute is considered as discerning, if it is an overlapping dimension, but the proportion of overlapping is tolerable based on user-defined parameter θ_1 $(0 < \theta_1 \leq 1)$.

The partial overlapping check based on proportions is irrelevant to point hyperboxes. In all the other cases 4th to 11th, the proportionality of overlap (propoverlap) is determined as follows:

$$propoverlap = max(\frac{o_p}{(w_a^{H_i} - v_a^{H_i})}, \frac{o_p}{(w_a^{H_j} - v_a^{H_j})}) \tag{5.3}$$

The amount of overlapping existing in each case is given by o_p. $propoverlap$ gives the maximum of proportionality of overlap in both hyperboxes, and it should be lesser than given θ_1 for an attribute to be included in discernibility matrix entry.

Algorithm 5.1 presents the structure for computing the crisp DM based on IDS. In Algorithm 5.1, for every pair of hyperboxes of different classes, an entry $M(H_i, H_j)$ is created by considering only those attributes over which no overlapping exists, or permissible partial overlapping exists.

The crisp DM construction through fuzzy hyperboxes is an approximation of crisp DM based on object space. Therefore, the reduct often computed through crisp DM is always a sub-reduct of the exact reduct; hence it is an approximate reduct for the original decision system.

The advantage of the proposed approach is that the discernibility entry preserves those important attributes which have the potential to discern most of the pair of objects from both hyperboxes. Hence, attributes with higher discerning power retained in M, thus paving the way for the construction of approximate reduct containing useful attributes.

Algorithm 5.1: Creating Crisp Discernibility Matrix

Input : HBS: Set of hyperboxes, θ_1: User-defined tolerance parameter, C^n: Set of conditional attributes
Output: M: Crisp Discernibility Matrix.

for *every* H_i *in* $|HBS|$ **do**
  for *every* H_j *in* $|HBS|$ **do**
    // Compute $M(H_i, H_j)$ for i^{th} hyperbox with each j^{th} hyperbox of different class labels
    if $d(H_i) \neq d(H_j)$ **then**
      for *each* a *in* C^n **do**
        if $OverlapInDim(H_i, H_j, a) == False$ **then**
          add($M(H_i, H_j)$,a);
        end
        $propoverlap = max(\frac{o_p}{(w_a^{H_i} - v_a^{H_i})}, \frac{o_p}{(w_a^{H_j} - v_a^{H_j})})$;
        if $OverlapInDim(H_i, H_j, a) == True$ *and* $propoverlap < \theta_1$ **then**
          add($M(H_i, H_j)$,a);
        end
      end
    end
  end
end
return M

5.3.3 Reduct Computation using Johnson's Reducer

In the last phase, Johnson's algorithm [144] is used to find a single reduct through crisp DM. Johnson's algorithm is given in Algorithm 5.2.

Johnson's algorithm is a greedy hill-climbing algorithm based on maximal discernibility heuristic (MDHeuristic). MDHeuristic is an estimation of the discernibility power of an attribute, and is equal to the number of DM entries containing the attribute. Johnson's algorithm is a sequential forward selection strategy based algorithm and starts with an empty set reduct. In each iteration, MDHeuristic computes for each attribute not already included in reduct. The best discerning attribute is included into the reduct, and the corresponding clauses containing the attribute are removed before proceeding to the next iteration. The removal of clauses is needed as the discerning pair of objects (in our case a pair of hyperboxes) require only a single attribute of the corresponding matrix entries.

Further, the removal of clauses reduces space complexity for successive iterations. The iteration continues till M becomes empty. After the end condition is reached the reduct obtained is returned by Johnson's algorithm.

M is an approximation of the crisp DM for the given dataset, the application of Johnson's algorithm on M results in an approximate reduct for the decision systems. Hence, checking the quality of the approximate reduct is one of the objectives of the experiment conducted in Section 5.5.

Algorithm 5.2: Finding Single Reduct using Johnson's Reducer

```
Input  : M: Crisp discernibility matrix, C^n: Set of conditional attributes
Output: Red: Approximate reduct
Red = ∅;
while M not empty do
    bestMD = 0;
    for each a in C^n − Red do
        R = MDHeuristic(a);
        if R > bestMD then
            bestMD = R;
            a^best = a;
        end
    end
    Red = Red ∪ {a^best} ;
    RemoveClauses(M,a^best);
end
return Red
```

5.4 Complexity Analysis of CDM-FMFRS Algorithm

This section shows the time and space complexity analysis of the proposed algorithm CDM-FMFRS. The following variables are used in the complexity analysis of CDM-FMFRS.

- $|U|$: the number of objects.
- $|HBS|$: the number of hyperboxes.
- $|C^n|$: the number of numeric conditional attribute.
- $|M|$: Size of discernibility matrix

Table 5.1 shows the time complexity of the proposed algorithm CDM-FMFRS. The procedure for IDS construction in CDM-FMFRS is as same as in FDM-FMFRS with a time complexity of $O(|U| * |HBS| * |C^n|)$. In Table 5.1, Algorithm 5.1 with steps 2-29 constructs the crisp DM based on IDS with a time complexity $O(|HBS|^2 * |C^n|)$ which is also theoretically equivalent to the construction of fuzzy DM in FDM-FMFRS. Algorithm 5.2 with steps 2-13 perform reduct computation based on Johnson reducer on crisp DM using SFS based control strategy with a time complexity of $O(|M| * |C^n|^2) = O(|HBS|^2 * |C^n|^2)$, since $|M| = O(|HBS|^2)$.

So, the total complexity of the proposed algorithm CDM-FMFRS is: $O(|U|*|HBS|*|C^n|)$ + $O(|HBS|^2 * |C^n|^2)$.

Theoretically, the space complexity of CDM-FMFRS is equivalent to FDM-FMFRS, i.e., $O(|U|*|C^n|) + O(|HBS|^2*|C^n|)$. But, as described in Section 5.2, practically, CDM-FMFRS space complexity is measured in terms of space utilization of crisp DM, which is $\frac{1}{8\times k}$ ('k' is the computer real-valued numbered bytes) of space utilization of fuzzy DM that forms the main advantage of CDM-FMFRS.

Table 5.1: Time Complexity Analysis of CDM-FMFRS Algorithm

Algorithm (phase)	**Steps in Algorithm**	**Time complexity**
Algorithm 3.1	2-29. Construction of IDS	$O(\|U\| * \|HBS\| * \|C^n\|)$
Algorithm 5.1	1-15. Construction of fuzzy DM	$O(\|HBS\|^2 * \|C^n\|)$
Algorithm 5.2	2-13. Reduct computation	$O(\|M\| * \|C^n\|^2) = O(\|HBS\|^2 * \|C^n\|^2)$

5.5 Experiment

The hardware configuration of the system used for experiments is CPU: Intel(R) i7-8500, Clock Speed: 3.40GHz × 6, RAM: 32 GB DDR4, OS: Ubuntu 18.04 LTS 64 bit and Software: Matlab R2017a. The detailed experimental evaluation is conducted on twenty benchmark numeric decision systems taken from UCI machine learning repository [21], the details are given in Table 5.2. The proposed algorithm FDM-FMFRS is implemented in the Matlab environment. In our experiments, we set the sensitive parameter γ value equal to 4, as recommended [63, 102]. And, based on the selected theta (θ) parameter in Chapter 3, we deduced that *theta* values of 0.3 and propoverlap θ_1 value of 0.1 are appropriate in the computation of CDM-FMFRS algorithm.

The performance of the proposed algorithm CDM-FMFRS is assessed by comparing it with FDM-FMFRS and recent state-of-the-art approaches developed for FRS reduct computation in 2018 and 2019 (same used in Chapter 4 comparative experiment) named as RMDPS [15], WRMDPS [15], FWARA [138] and PARA [72]. Furthermore, these comparative approaches (RMDPS, WRMDPS, FWARA and PARA) follow their own fuzzy model with t-norm, t-

Table 5.2: Benchmark Datasets

Dataset	Attributes	Objects	Class
Ionosphere	32	351	2
Vehicle	18	846	4
Segment	16	2310	2
Steel	27	1941	7
Ozone Layer	72	1848	2
Page	10	5472	5
Robot	24	5456	4
Waveform2	40	5000	3
Texture	40	5500	11
Thyroid	21	7200	3
Gamma	10	19020	2
Satimage	36	6435	6
Ring	20	7400	2
Musk2	166	6598	2
Shuttle	9	57999	7
Sensorless	48	58509	11
MiniBooNE	50	129596	2
Winnipeg	174	325834	7
Susy	18	5000000	2
Hepmass	29	(50000)10500000	2
Swarm Behaviour	2400	24017	2

conorm and fuzzy similarity relations as given in the respective publications and experiments are conducted in the same environment stated above. The comparative experiments are conducted in the same system using Matlab environment. The performance of CDM-FMFRS is examined through a comparative evaluation with respect to the following objectives:

1. Evaluate quality of approximate reduct through Gamma measure.
2. Comparative analysis of proposed approach in construction of different classifiers through ten-fold cross-validation (10-FCV).
3. Evaluate the performance on big datasets in achieving increased scalability.

5.5.1 Evaluating Quality of Reduct Computed by Proposed Approach

Reduct computation in CDM-FMFRS is based on a discernibility matrix construction in the hyperbox space. Since crisp DM on IDS is a transformation of fuzzy DM on IDS, theoretically, it results in an approximate reduct. Hence, some information loss is also present naturally.

The details of Gamma measure are precisely the same as followed in Chapter 4 on page number 61.

Table 5.3 contains the resulting gamma value and reduct length by applying the proposed algorithm as well as the compared algorithms on the entire dataset. Also, Table 5.3 represents the gamma measure obtained from the unreduced decision system (mention as 'UNRED' in Table 5.3) to validate the relevance of resulted reducts through checking whether the obtained reduct is satisfying or reaching near to (UNRED) gamma measure or not.

Table 5.3 reports the gamma value for only eleven datasets out of twenty benchmark datasets due to exceeding the memory limit while processing the GKFRS.

Table 5.3: Relevance of CDM-FMFRS reduct through Gamma measure

Datsets	**Gamma Meausre**						
	UNRED	**CDM-FMFRS**	**FDM-FMFRS**	**RMDPS**	**WRMDPS**	**FRAWA**	**PARA**
Ionosphere	0.99	0.99	0.98	0.99	0.99	0.99	0.99
Segment	0.98	0.98	0.94	0.98	0.98	0.98	0.96
Steel	0.98	0.98	0.94	0.98	0.98	0.98	0.98
Vehicle	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Ozone	1	1	0.99	1	1	1	1
Page	0.87	0.87	0.85	0.87	0.87	0.87	0.87
Texture	0.99	0.99	0.94	0.99	0.99	0.99	0.93
Waveform2	1	1	1	1	1	1	1
Robot	0.97	0.97	0.90	0.97	0.97	0.97	0.97
Satimage	0.99	0.99	0.98	0.99	0.99	0.99	0.98
Ring	1	1	1	1	1	1	1
Datsets	**Reduct Length**						
	UNRED	**CDM-FMFRS**	**FDM-FMFRS**	**RMDPS**	**WRMDPS**	**FRAWA**	**PARA**
Ionosphere	32	13	7	27	27	31	18
Segment	16	15	9	15	15	14	10
Steel	27	22	11	21	21	18	15
Vehicle	18	16	15	18	18	17	14
Ozone	72	46	9	39	42	54	29
Page	10	8	8	10	10	10	9
Texture	41	20	8	37	37	37	8
Waveform2	40	40	13	21	22	40	24
Robot	24	24	13	24	24	24	24
Satimage	36	36	14	36	36	36	14
Ring	20	20	17	20	20	20	18

Analysis of Results

In Table 5.3, it is observed that CDM-FMFRS have achieved the same gamma value as obtained by UNRED satisfying the required reduct property fully in all datasets.

It can also observe that the size of reduct for CDM-FMFRS is larger than FDM-FMFRS for all datasets except Page dataset (in page, all are giving full attribute size). CDM-FMFRS simply returns as a super-reduct of FDM-FMFRS as it achieves full gamma value. Due to information loss in crisp DM construction, there is a sparsity in crisp DM results in a larger reduct size.

Section 5.5.2 explores the relevance of obtained approximate reduct of the FDM-FMFRS in achieving the construction of the classification learning model, which is the primary objective of the feature subset selection. Moreover, the comparative analysis with reduct length and computational time will be elaborated as part of Section 5.5.2 using tenfold cross-validation.

5.5.2 Relevance of the Proposed Approach in Construction of Classifiers

This section contains the comparative experiments conducted among algorithms for reduct computation, i.e., CDM-FMFRS and FDM-FMFRS, RMDPS [15], WRMDPS [15], FWARA [138] and PARA [72]. The relevance of reduct in inducing a classification model is studied through ten-fold cross-validation (10-FCV) experiments. In each iteration, one fold is preserved for the testing data, and the remaining nine folds are used for training data. A reduct algorithm is applied to the training data. So, based on the reduct that is obtained, the classification model is constructed for comparison. The classification accuracy of the resulting model is evaluated based on the test data.

Two different classifier models are used, namely CART and kNN with default options, and for kNN experiments, k is taken as 3, and our proposed kNN-FMNN classifier (Chapter 3) is also employed for inducing classification. The classification experiments are conducted on the training data (mentioned under the column 'UNRED') to assess the relevance of the reduct in preserving the classifiability achieved by all attributes.

Table 5.4, Table 5.5 and Table 5.6 present the results of the 10-FCV experiment for classification accuracies with CART, kNN, and kNN-FMNN respectively. Similarly, Table 5.7 and Table 5.8 illustrates the reduct length and computational time of the algorithms. Fig. 5.1, Fig. 5.2, Fig. 5.3, Fig. 5.4 and Fig. 5.5 depict the box-plot representation of Table 5.4, Table 5.5, Table 5.6, Table 5.7 and Table 5.8 respectively. The results reported for FDM-FMFRS and other compared algorithms are as same as given in Chapter 4 (Section 4.7.2) and are reproduced here for each comprehension of comparative analysis with the proposed algorithm CDM-FMFRS.

The detailed student's paired t-test analysis and how the values are represented in Tables 5.4, 5.5, 5.6, 5.7 and 5.8 are precisely the same as followed in Chapter 4 on page number 62.

The last three lines in each Table 5.4, 5.5, 5.6, 5.7 and 5.8 correspond to **Average (NOD)**, **CAverage**, and **Lose/Win/Tie**. It can be observed that the datasets over which an algorithm is executing vary from one to another. Hence, the average of individual mean values is reported in two forms. Average (NOD) corresponds to the average value obtained by an algorithm on datasets where it could be evaluated along with reporting the number of datasets (NOD) involved in brackets. CAverage value depicts the average of the individual mean obtained by restricting to only those datasets in which all algorithms could be evaluated. For the comparative analysis, CAverage plays an important role. The last line indicates the count of the number of statistically loss('-'), better('+'), and equivalent('o') for each algorithm in comparison with the proposed CDM-FMFRS.

Note: The '*' sign in Tables 5.4, 5.5, 5.6, 5.7 and 5.8 shows the corresponding algorithm is intractable to a particular dataset to compute the reduct due to insufficient memory. And, '#' sign represents the scenario of non-termination of the code even after several hours of computation.

In Figures 5.1, 5.2, 5.3, 5.4 and 5.5 , the range of Y-axis varies based on obtained results in each dataset. For large datasets, as the results are available only for CDM-FMFRS and FDM-FMFRS algorithms, Figures are respectively given in Figure (b) part.

Table 5.4: Classification Accuracies Results (%) with CART in 10-FCV

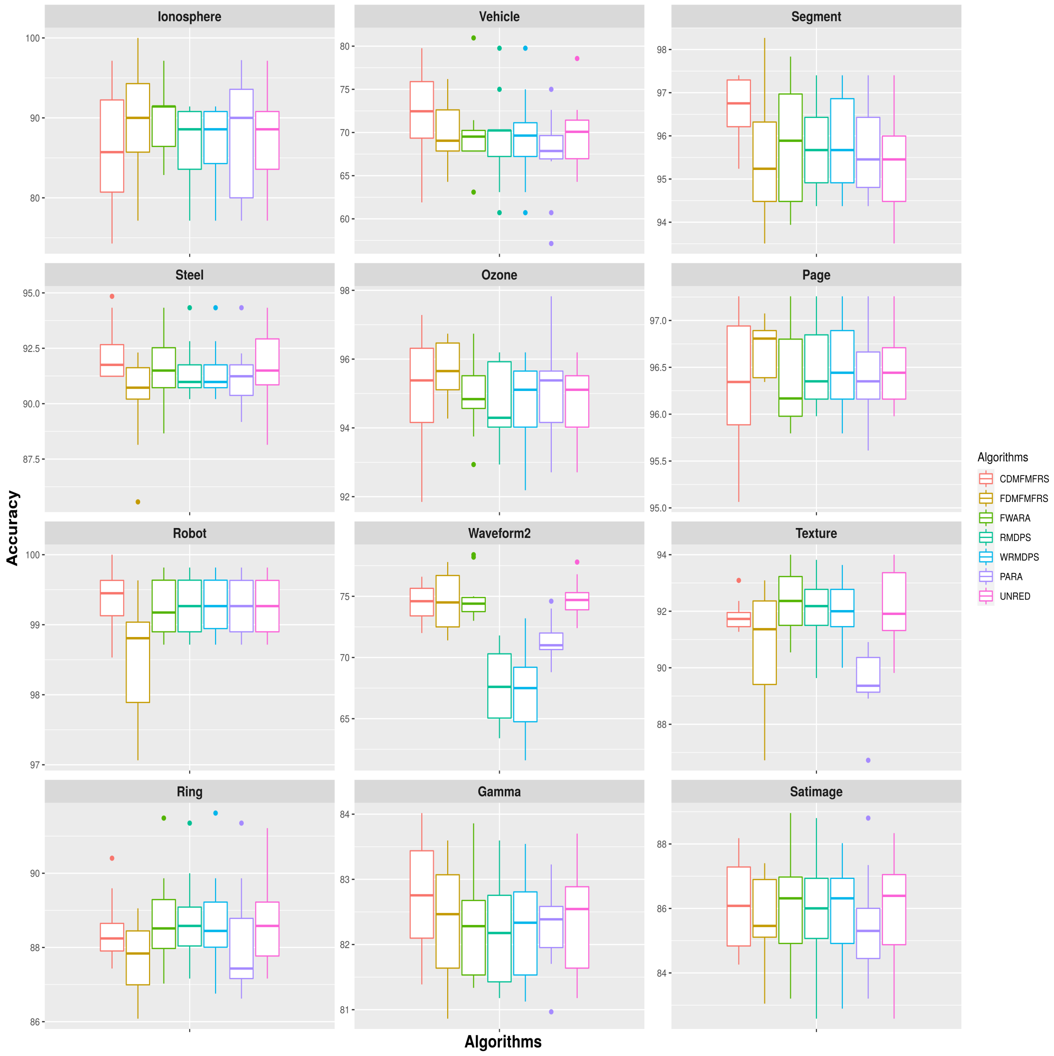
Datasets	CDM-FMFRS	FDM-FMFRS		RMDPS		WRMDPS		FWARA		PARA		UNRED	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	86.04 ± 7.31	88.87 ± 7.57	0.41^o	86.60 ± 5.06	0.84^o	86.89 ± 5.09	0.77^o	89.73 ± 4.72	0.20^o	87.15 ± 7.78	0.75^o	87.46 ± 5.59	0.63^o
Vehicle	72.11 ± 5.16	69.83 ± 3.76	0.27^o	69.51 ± 5.42	0.29^o	69.51 ± 5.45	0.29^o	69.86 ± 4.52	0.31^o	67.37 ± 5.21	0.06^o	69.85 ± 4.15	0.29^o
Segment	96.62 ± 0.76	95.45 ± 1.57	0.05^-	95.76 ± 1.00	0.04^-	95.80 ± 1.12	0.07^o	95.84 ± 1.42	0.14^o	95.63 ± 1.09	0.03^-	95.28 ± 1.27	0.01^-
Steel	92.27 ± 1.33	90.26 ± 2.01	0.02^-	91.50 ± 1.25	0.20^o	91.50 ± 1.25	0.20^o	91.45 ± 1.64	0.23^o	91.24 ± 1.45	0.12^o	91.60 ± 1.88	0.37^o
Ozone	95.12 ± 1.67	95.62 ± 0.90	0.42^o	94.65 ± 1.23	0.48^o	94.71 ± 1.40	0.55^o	94.97 ± 1.20	0.82^o	94.98 ± 1.50	0.84^o	94.65 ± 1.18	0.47^o
Page	96.36 ± 0.71	96.71 ± 0.30	0.17^o	96.51 ± 0.50	0.60^o	96.49 ± 0.49	0.65^o	96.35 ± 0.52	0.95^o	96.44 ± 0.53	0.80^o	96.45 ± 0.42	0.73^o
Robot	99.36 ± 0.43	98.53 ± 0.82	0.01^-	99.27 ± 0.44	0.64^o	99.28 ± 0.43	0.71^o	99.25 ± 0.45	0.58^o	99.25 ± 0.42	0.57^o	99.25 ± 0.42	0.57^o
Waveform2	74.52 ± 1.54	74.66 ± 2.45	0.88^o	67.70 ± 2.97	0.00^-	67.10 ± 3.79	0.00^-	74.94 ± 1.86	0.59^o	71.44 ± 1.75	0.00^-	74.82 ± 1.58	0.67^o
Texture	91.82 ± 0.56	90.82 ± 2.11	0.16^o	92.07 ± 1.22	0.56^o	92.02 ± 1.14	0.63^o	92.33 ± 1.17	0.23^o	89.47 ± 1.20	0.00^-	92.09 ± 1.37	0.57^o
Ring	88.46 ± 0.91	87.70 ± 0.94	0.08^o	88.78 ± 1.18	0.50^o	88.69 ± 1.36	0.66^o	88.77 ± 1.27	0.54^o	88.12 ± 1.55	0.56^o	88.73 ± 1.27	0.59^o
Gamma	82.70 ± 0.94	82.36 ± 0.93	0.43^o	82.20 ± 0.84	0.22^o	82.22 ± 0.82	0.25^o	82.27 ± 0.81	0.29^o	82.29 ± 0.67	0.27^o	82.34 ± 0.85	0.38^o
Satimage	86.14 ± 1.44	85.75 ± 1.38	0.54^o	85.98 ± 1.78	0.83^o	86.00 ± 1.61	0.84^o	85.94 ± 1.80	0.78^o	85.55 ± 1.61	0.40^o	85.98 ± 1.70	0.83^o
Shuttle	98.68 ± 1.12	99.96 ± 0.02	0.00^+	*		*		*		99.95 ± 0.02	0.27^o	99.96 ± 0.02	0.00^+
Musk2	96.88 ± 0.52	97.74 ± 0.43	0.20^o	*		*		*		#		97.03 ± 0.36	0.46^o
Sensorless	95.09 ± 2.05	96.69 ± 2.83	0.17^o	*		*		*		#		98.38 ± 0.19	0.00^+
MiniBooNE	89.53 ± 0.29	88.75 ± 0.96	0.02^-	*		*		*		#		89.58 ± 0.28	0.69^o
Winnipeg	98.92 ± 0.05	98.46 ± 0.10	0.00^-	*		*		*		#		98.92 ± 0.08	1.00^o
Average (NOD)	90.74 (17)	90.72 (17)		87.59 (12)		87.56 (12)		88.47 (12)		89.16 (13)		91.28 (17)	
CAverage$^{\$}$	88.46	88.08		87.59		87.56		88.47		87.41		88.29	
Lose/Win/Tie		5/1/11		2/0/10		1/0/11		0/0/12		3/0/10		1/2/14	

NOD: Number of datasets over which the average is computed (indicated in bracket).

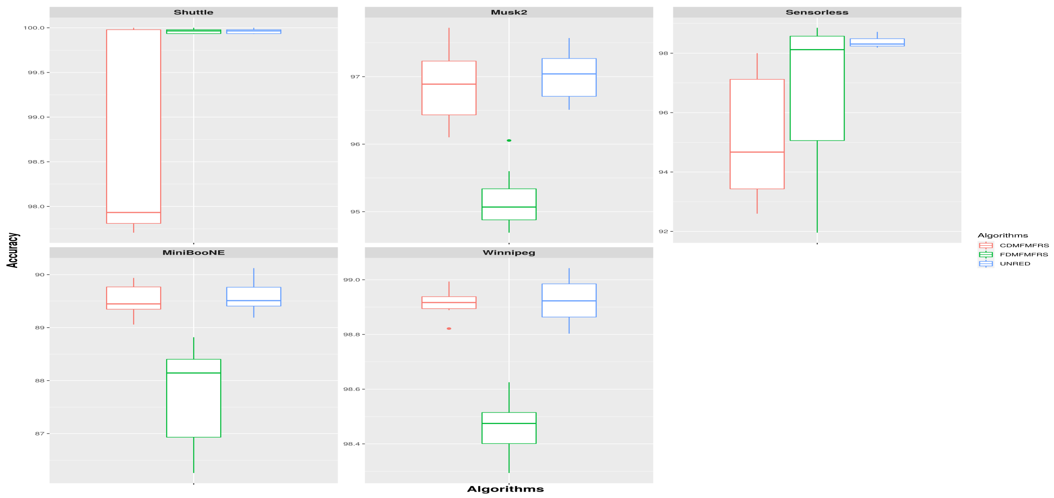
$: Average mean value over 7 datasets where all algorithms executed.

***** represents non-executable due to memory overflow.

represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



(b) Datasets having classification results by FDM-FMFRS, CDM-FMFRS and UNRED

Figure 5.1: Boxplot for Classification Accuracies Results with CART of Table 5.4

Table 5.5: Classification Accuracies Results (%) with kNN (k=3) in 10-FCV

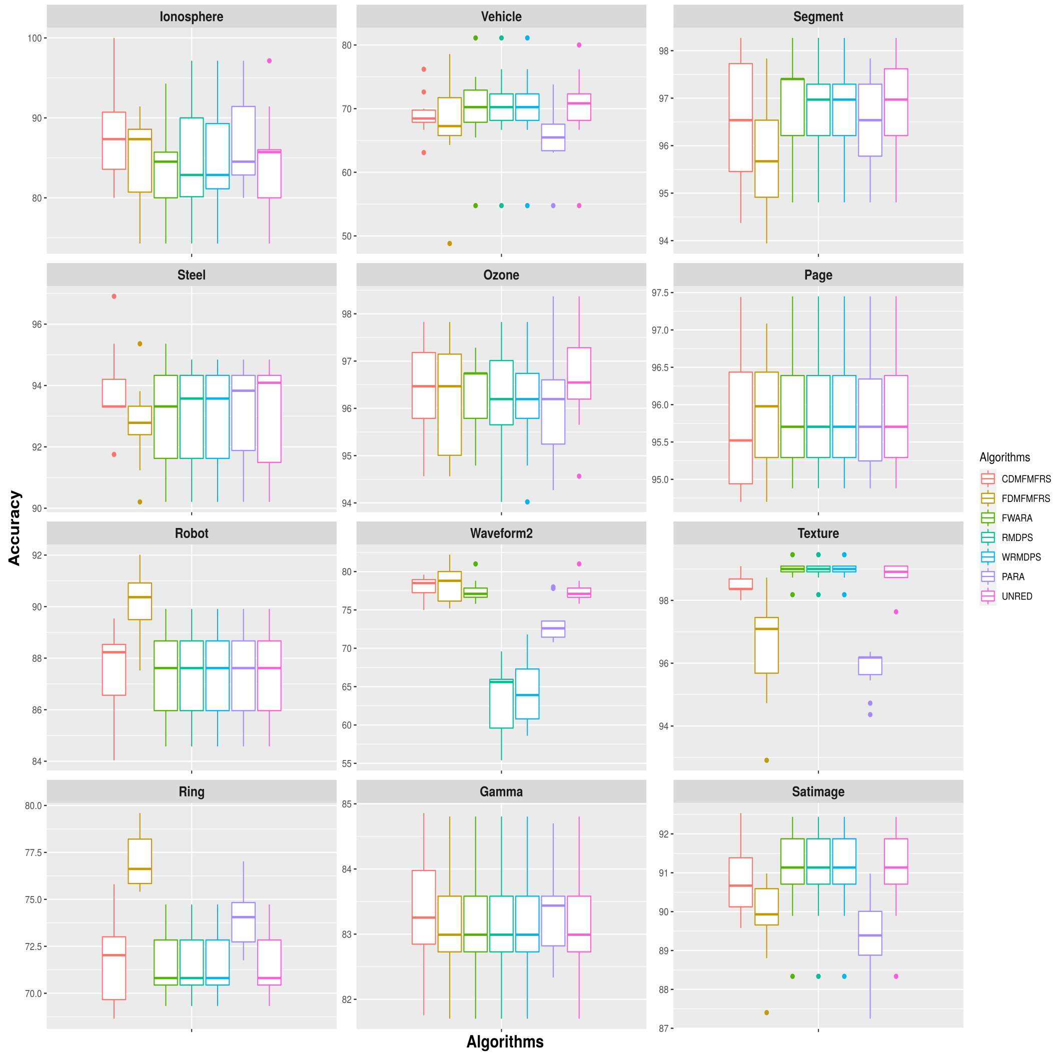
Datasets	CDM-FMFRS	FDM-FMFRS		RMDPS		WRMDPS		FWARA		PARA		UNRED	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	87.75 ± 5.71	84.90 ± 5.40	0.27^o	84.91 ± 7.22	0.34^o	84.91 ± 7.09	0.34^o	83.19 ± 6.24	0.11^o	86.62 ± 5.85	0.67^o	84.04 ± 7.16	0.22^o
Vehicle	69.02 ± 3.50	67.56 ± 8.04	0.60^o	70.02 ± 6.83	0.69^o	70.02 ± 6.83	0.69^o	69.66 ± 6.86	0.80^o	65.24 ± 4.85	0.06^o	70.02 ± 6.65	0.68^o
Segment	96.49 ± 1.38	95.80 ± 1.28	0.26^o	96.75 ± 1.03	0.64^o	96.75 ± 1.03	0.64^o	96.88 ± 1.08	0.49^o	96.49 ± 1.05	1.00^o	96.80 ± 1.06	0.59^o
Steel	93.71 ± 1.55	92.79 ± 1.40	0.18^o	93.04 ± 1.77	0.38^o	93.04 ± 1.77	0.38^o	93.10 ± 1.74	0.41^o	93.10 ± 1.60	0.39^o	93.15 ± 1.74	0.45^o
Ozone	96.43 ± 0.97	96.16 ± 1.21	0.60^o	96.11 ± 1.17	0.52^o	96.22 ± 1.20	0.67^o	96.33 ± 0.87	0.81^o	96.06 ± 1.24	0.47^o	96.65 ± 1.11	0.64^o
Page	95.74 ± 0.93	95.89 ± 0.82	0.72^o	95.85 ± 0.83	0.79^o	95.85 ± 0.83	0.79^o	95.85 ± 0.83	0.79^o	95.83 ± 0.82	0.82^o	95.85 ± 0.83	0.79^o
Robot	87.50 ± 1.66	90.17 ± 1.27	0.00^+	87.41 ± 1.86	0.91^o	87.41 ± 1.86	0.91^o	87.41 ± 1.86	0.91^o	87.41 ± 1.86	0.91^o	87.41 ± 1.86	0.91^o
Waveform2	78.00 ± 1.40	78.54 ± 2.53	0.56^o	63.38 ± 4.47	0.00^-	64.44 ± 4.81	0.00^-	77.50 ± 1.49	0.45^o	73.30 ± 2.60	0.00^-	77.50 ± 1.49	0.45^o
Texture	98.47 ± 0.31	96.55 ± 1.73	0.00^-	98.98 ± 0.37	0.00^+	98.98 ± 0.37	0.00^+	98.98 ± 0.37	0.00^+	95.82 ± 0.72	0.00^-	98.80 ± 0.44	0.07^o
Ring	71.78 ± 2.32	77.12 ± 1.60	0.00^+	71.62 ± 1.82	0.86^o	71.62 ± 1.82	0.86^o	71.62 ± 1.82	0.86^o	74.12 ± 1.75	0.02^+	71.62 ± 1.82	0.86^o
Gamma	83.28 ± 0.97	83.13 ± 0.91	0.74^o	83.13 ± 0.91	0.74^o	83.13 ± 0.91	0.74^o	83.13 ± 0.91	0.74^o	83.36 ± 0.73	0.83^o	83.13 ± 0.91	0.74^o
Satimage	90.82 ± 0.97	89.85 ± 1.09	0.05^-	91.05 ± 1.25	0.65^o	91.05 ± 1.25	0.65^o	91.05 ± 1.25	0.65^o	89.40 ± 1.09	0.01^-	91.05 ± 1.25	0.65^o
Shuttle	98.36 ± 1.34	99.92 ± 0.02	0.00^+	*		*		*		99.90 ± 0.03	0.09^o	99.91 ± 0.03	0.00^+
Musk2	96.47 ± 0.53	95.15 ± 0.82	0.00^-	*		*		*		#		96.83 ± 0.83	0.26^o
Sensorless	90.90 ± 2.59	97.57 ± 3.61	0.01^+	*		*		*		#		99.02 ± 0.19	0.00^+
MiniBooNE	92.27 ± 0.21	90.36 ± 0.92	0.00^-	*		*		*		#		92.19 ± 0.15	0.33^o
Winnipeg	99.59 ± 0.04	99.62 ± 0.09	0.38^o	*		*		*		#		99.60 ± 0.04	0.50^o
Average (NOD)	89.54 (17)	90.17 (17)		85.98 (12)		86.08 (12)		87.02 (12)		87.89 (13)		90.39 (17)	
CAverage$^{\$}$	87.41	87.34		85.98		86.08		87.02		86.37		87.13	
Lose/Win/Tie		4/4/9		1/1/10		1/1/10		0/1/11		3/1/9		0/2/15	

NOD: Number of datasets over which the average is computed (indicated in bracket).

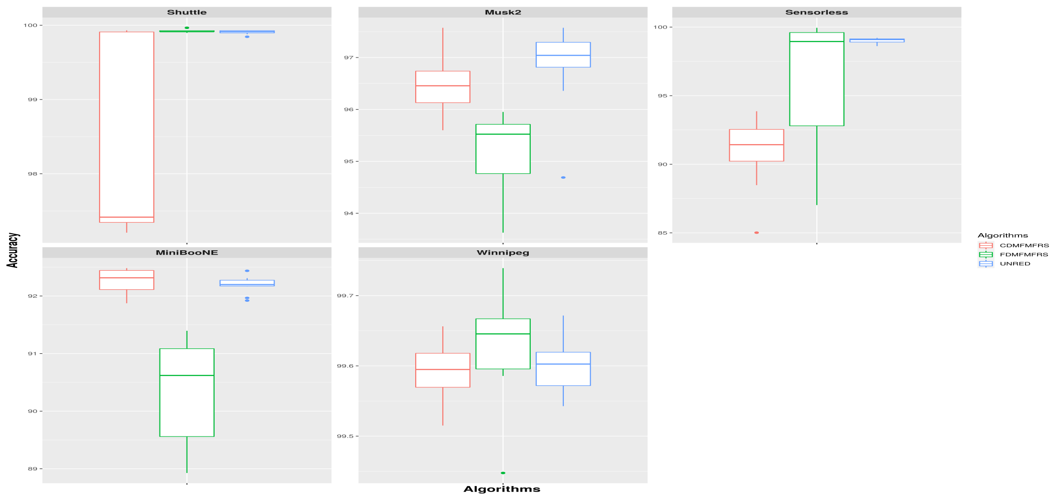
$: Average mean value over 7 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



(b) Datasets having classification results by FDM-FMFRS, CDM-FMFRS and UNRED

Figure 5.2: Boxplot for Classification Accuracies Results with kNN of Table 5.5

Table 5.6: Classification Accuracies Results (%) with kNN-FMNN in 10-FCV

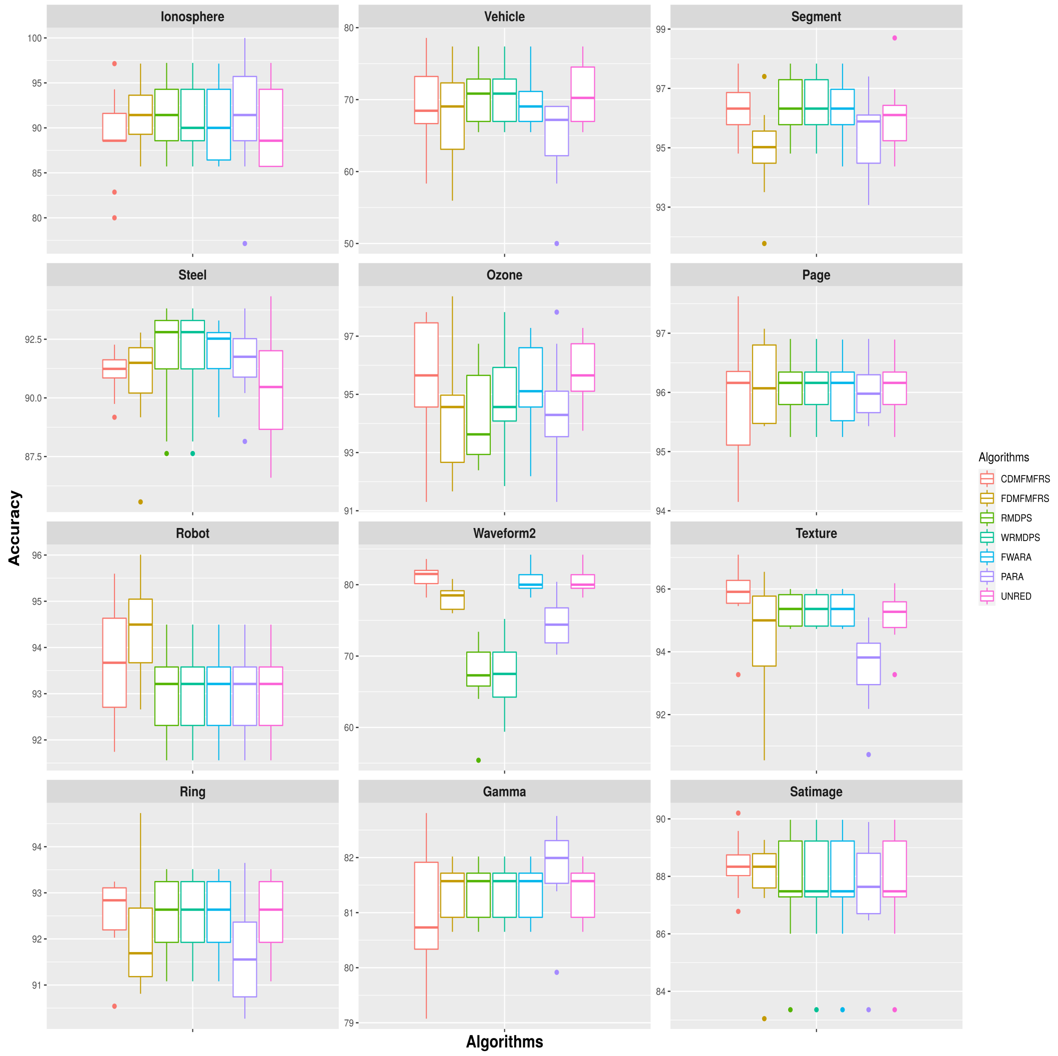
Datasets	**CDM-FMFRS**	**FDM-FMFRS**		**RMDPS**		**WRMDPS**		**FWARA**		**PARA**		**UNRED**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	89.17 ± 5.02	91.74 ± 3.67	0.21^{o}	91.44 ± 4.27	0.29^{o}	91.44 ± 4.05	0.28^{o}	90.59 ± 4.29	0.50^{o}	90.87 ± 6.58	0.52^{o}	90.29 ± 5.09	0.62^{o}
Vehicle	69.11 ± 5.76	68.14 ± 6.98	0.74^{o}	70.67 ± 4.43	0.51^{o}	70.67 ± 4.43	0.51^{o}	69.71 ± 3.59	0.78^{o}	64.39 ± 6.32	0.10^{o}	70.77 ± 4.63	0.49^{o}
Segment	96.32 ± 0.94	94.89 ± 1.51	0.02^{-}	96.45 ± 1.06	0.77^{o}	96.45 ± 1.06	0.77^{o}	96.32 ± 1.05	1.00^{o}	95.45 ± 1.28	0.10^{o}	96.02 ± 1.24	0.55^{o}
Steel	91.04 ± 0.94	90.73 ± 2.14	0.68^{o}	91.81 ± 2.27	0.33^{o}	91.81 ± 2.27	0.33^{o}	91.81 ± 1.42	0.17^{o}	91.50 ± 1.58	0.44^{o}	90.37 ± 2.63	0.46^{o}
Ozone	95.40 ± 2.27	94.17 ± 1.98	0.21^{o}	94.16 ± 1.62	0.18^{o}	94.92 ± 1.79	0.61^{o}	95.31 ± 1.52	0.92^{o}	94.43 ± 1.92	0.32^{o}	95.84 ± 1.15	0.59^{o}
Page	95.89 ± 1.04	96.14 ± 0.67	0.52^{o}	96.11 ± 0.56	0.56^{o}	96.11 ± 0.56	0.56^{o}	96.03 ± 0.59	0.70^{o}	96.05 ± 0.49	0.66^{o}	96.09 ± 0.53	0.59^{o}
Robot	93.64 ± 1.24	94.39 ± 1.04	0.16^{o}	93.04 ± 0.97	0.24^{o}	93.04 ± 0.97	0.24^{o}	93.04 ± 0.97	0.24^{o}	93.04 ± 0.97	0.24^{o}	93.04 ± 0.97	0.24^{o}
Waveform2	81.06 ± 1.78	78.18 ± 1.66	0.00^{-}	67.04 ± 5.02	0.00^{-}	67.56 ± 4.79	0.00^{-}	80.58 ± 1.88	0.56^{o}	74.60 ± 3.42	0.00^{-}	80.58 ± 1.88	0.56^{o}
Texture	95.80 ± 1.03	94.38 ± 1.93	0.05^{-}	95.36 ± 0.53	0.25^{o}	95.36 ± 0.53	0.25^{o}	95.36 ± 0.53	0.25^{o}	93.55 ± 1.34	0.00^{-}	95.11 ± 0.82	0.11^{o}
Ring	92.41 ± 1.04	92.12 ± 1.31	0.60^{o}	92.47 ± 0.86	0.88^{o}	92.47 ± 0.86	0.88^{o}	92.47 ± 0.86	0.88^{o}	91.69 ± 1.16	0.16^{o}	92.47 ± 0.86	0.88^{o}
Gamma	80.95 ± 1.15	81.38 ± 0.49	0.30^{o}	81.38 ± 0.49	0.30^{o}	81.38 ± 0.49	0.30^{o}	81.38 ± 0.49	0.30^{o}	81.82 ± 0.81	0.06^{o}	81.38 ± 0.49	0.30^{o}
Satimage	88.39 ± 1.01	87.85 ± 1.82	0.42^{o}	87.61 ± 1.99	0.28^{o}	87.61 ± 1.99	0.28^{o}	87.61 ± 1.99	0.28^{o}	87.54 ± 1.87	0.22^{o}	87.61 ± 1.99	0.28^{o}
Shuttle	92.96 ± 6.01	99.94 ± 0.03	0.00^{+}	*		*		*		99.91 ± 0.04	0.07^{o}	99.92 ± 0.03	0.00^{+}
Musk2	96.21 ± 0.79	94.12 ± 0.76	0.00^{-}	*		*		*		#		96.36 ± 0.65	0.65^{o}
Sensorless	90.92 ± 2.62	85.33 ± 3.31	0.00^{-}	*		*		*		#		94.82 ± 0.31	0.00^{+}
MiniBooNE	89.73 ± 0.18	88.51 ± 0.83	0.00^{-}	*		*		*		#		89.71 ± 0.21	0.83^{o}
Winnipeg	98.09 ± 0.04	98.99 ± 0.13	0.00^{+}	*		*		*		#		98.07 ± 0.11	0.00^{-}
Average (NOD)	90.41 (17)	90.34 (17)		87.92 (12)		88.02 (12)		89.21 (12)		89.22 (13)		91.31 (17)	
CAverage$^{\$}$	89.09	88.69		87.92		88.02		89.21		87.86		88.96	
Lose/Win/Tie		6/2/9		1/0/11		1/0/11		0/0/12		2/0/11		1/2/14	

NOD: Number of datasets over which the average is computed (indicated in bracket).

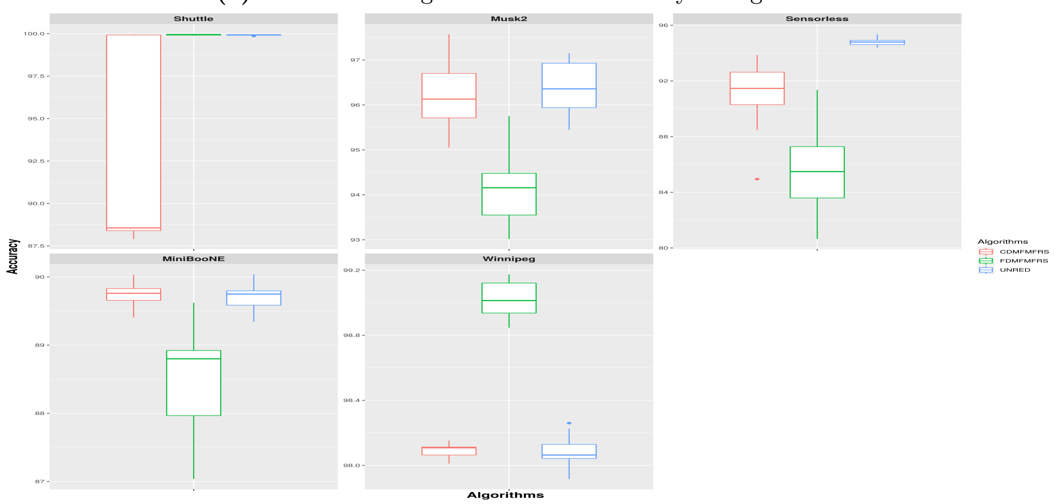
$: Average mean value over 7 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



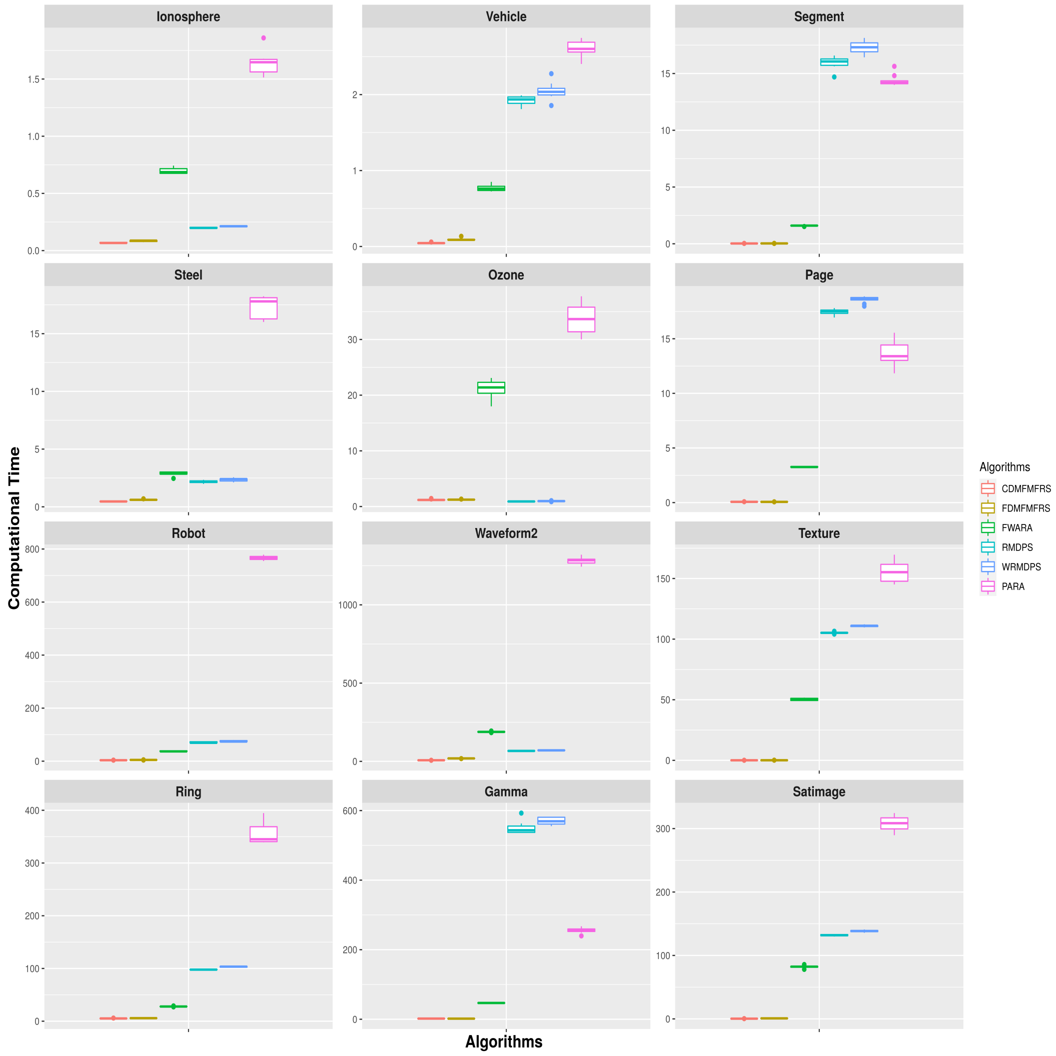
(b) Datasets having classification results by FDM-FMFRS, CDM-FMFRS and UNRED

Figure 5.3: Boxplot for Classification Accuracies Results with kNN-FMNN of Table 5.6

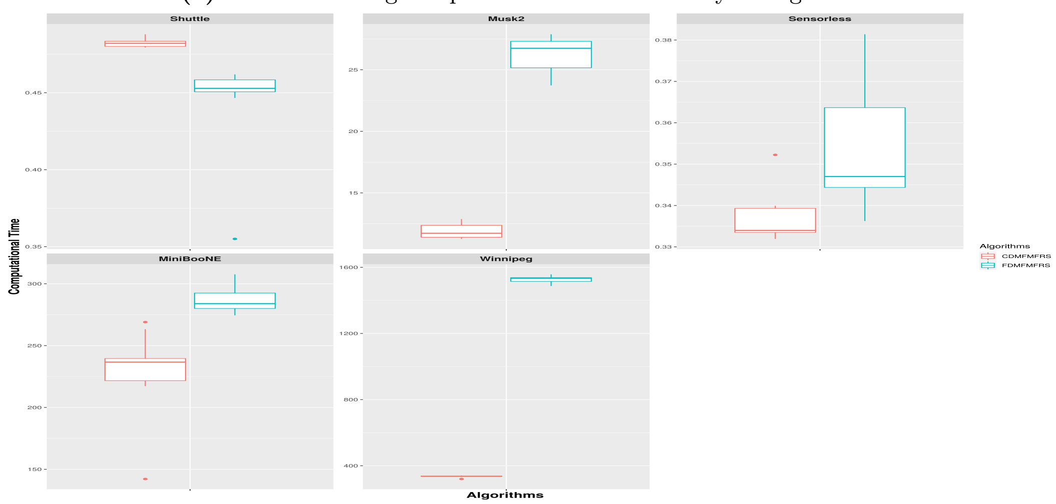
Table 5.7: Computational Times Results (in seconds) in 10-FCV

Datasets	**CDM-FMFRS**	**FDM-FMFRS**		**RMDPS**		**WRMDPS**		**FWARA**		**PARA**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	0.07 ± 0.00	0.08 ± 0.01	0.00^-	0.20 ± 0.01	0.00^-	0.21 ± 0.00	0.00^-	0.70 ± 0.03	0.00^-	1.64 ± 0.10	0.00^-
Vehicle	0.05 ± 0.01	0.09 ± 0.02	0.00^-	1.92 ± 0.06	0.00^-	2.05 ± 0.11	0.00^-	0.77 ± 0.04	0.00^-	2.60 ± 0.12	0.00^-
Segment	0.02 ± 0.00	0.03 ± 0.00	0.00^-	15.96 ± 0.54	0.00^-	17.29 ± 0.55	0.00^-	1.60 ± 0.04	0.00^-	14.38 ± 0.50	0.00^-
Steel	0.46 ± 0.02	0.61 ± 0.04	0.00^-	2.17 ± 0.11	0.00^-	2.35 ± 0.14	0.00^-	2.88 ± 0.18	0.00^-	17.32 ± 0.95	0.00^-
Ozone	1.22 ± 0.10	1.24 ± 0.04	0.45^o	0.90 ± 0.04	0.00^-	0.97 ± 0.05	0.00^-	21.12 ± 1.57	0.00^-	33.69 ± 2.72	0.00^-
Page	0.06 ± 0.00	0.07 ± 0.00	0.01^-	17.47 ± 0.26	0.00^-	18.58 ± 0.29	0.00^-	3.27 ± 0.03	0.00^-	13.70 ± 1.16	0.00^-
Robot	3.59 ± 0.16	4.68 ± 0.27	0.00^-	70.82 ± 2.78	0.00^-	75.37 ± 2.64	0.00^-	37.65 ± 1.72	0.00^-	765.91 ± 8.04	0.00^-
Waveform2	7.11 ± 0.04	18.86 ± 0.48	0.00^-	66.76 ± 0.36	0.00^-	70.59 ± 0.84	0.00^-	188.27 ± 2.94	0.00^-	1279.60 ± 22.61	0.00^-
Texture	0.06 ± 0.00	0.08 ± 0.02	0.00^-	105.28 ± 0.65	0.00^-	110.88 ± 0.72	0.00^-	50.34 ± 0.98	0.00^-	155.37 ± 8.53	0.00^-
Ring	5.28 ± 0.34	5.75 ± 0.05	0.00^-	98.00 ± 0.67	0.00^-	103.34 ± 0.85	0.00^-	27.87 ± 0.49	0.00^-	355.59 ± 21.58	0.00^-
Gamma	1.88 ± 0.11	1.77 ± 0.04	0.01^+	549.37 ± 17.86	0.00^-	569.81 ± 10.67	0.00^-	46.83 ± 0.28	0.00^-	255.27 ± 7.46	0.00^-
Satimage	0.37 ± 0.01	0.85 ± 0.04	0.00^-	131.86 ± 1.18	0.00^-	138.36 ± 1.64	0.00^-	82.40 ± 2.13	0.00^-	307.66 ± 12.38	0.00^-
Shuttle	0.48 ± 0.00	0.44 ± 0.03	0.00^+	*		*		*		1243.4 ± 0.06	0.00^-
Musk2	11.92 ± 0.62	26.22 ± 1.38	0.00^-	*		*		*		#	
Sensorless	0.34 ± 0.01	0.35 ± 0.02	0.00^-	*		*		*		#	
MiniBooNE	229.60 ± 34.90	287.48 ± 11.21	0.00^-	*		*		*		#	
Winnipeg	335.42 ± 5.96	1656.66 ± 34.79	0.00^-	*		*		*		#	
Average (NOD)	35.15 (17)	117.38 (17)		88.65 (12)		92.59 (12)		38.49 (12)		321.44 (13)	
CAverage$^{\$}$	1.68	2.85		88.65		92.59		38.49		266.88	
Lose/Win/Tie		14/2/1		12/0/0		12/0/0		12/0/0		13/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 7 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.



(a) Datasets having computational time results by all algorithms



(b) Datasets having computational time results by FDM-FMFRS and CDM-FMFRS

Figure 5.4: Boxplot for Computational Time Results of Table 5.7

Table 5.8: Reduct Length Results in 10-FCV

Datasets	**CDM-FMFRS**	**FDM-FMFRS**		**RMDPS**		**WRMDPS**		**FWARA**		**PARA**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	12.10 ± 1.73	6.60 ± 0.70	0.00^{+}	26.70 ± 1.06	0.00^{-}	27.60 ± 1.35	0.00^{-}	30.80 ± 0.79	0.00^{-}	16.70 ± 0.82	0.00^{-}
Vehicle	16.10 ± 0.57	12.90 ± 0.99	0.00^{+}	17.80 ± 0.42	0.00^{-}	17.80 ± 0.42	0.00^{-}	16.70 ± 0.48	0.02^{-}	8.60 ± 0.52	0.00^{+}
Segment	7.50 ± 1.51	8.10 ± 0.88	0.29^{o}	15.00 ± 0.00	0.00^{-}	15.00 ± 0.00	0.00^{-}	14.10 ± 0.32	0.00^{-}	9.20 ± 0.42	0.00^{-}
Steel	22.00 ± 1.05	12.80 ± 1.62	0.00^{+}	20.00 ± 0.94	0.00^{+}	20.00 ± 0.94	0.00^{+}	17.20 ± 0.63	0.00^{+}	14.90 ± 0.32	0.00^{+}
Ozone	43.80 ± 2.20	9.60 ± 0.84	0.00^{+}	37.40 ± 1.07	0.00^{+}	39.40 ± 1.78	0.00^{+}	50.90 ± 1.91	0.00^{-}	28.10 ± 0.88	0.00^{+}
Page	8.40 ± 0.97	7.50 ± 1.27	0.09^{o}	9.90 ± 0.32	0.00^{-}	9.90 ± 0.32	0.00^{-}	9.90 ± 0.32	0.00^{-}	8.80 ± 0.42	0.25^{o}
Robot	24.00 ± 0.00	12.50 ± 1.51	0.00^{+}	24.00 ± 0.00	1.00^{o}	24.00 ± 0.00	1.00^{o}	24.00 ± 0.00	1.00^{o}	24.00 ± 0.00	1.00^{o}
Waveform2	40.00 ± 0.00	13.00 ± 0.00	0.00^{+}	21.20 ± 0.42	0.00^{+}	21.60 ± 0.70	0.00^{+}	40.00 ± 0.00	1.00^{o}	23.90 ± 0.57	0.00^{+}
Texture	18.70 ± 1.89	8.60 ± 1.26	0.00^{+}	37.00 ± 0.00	0.00^{-}	37.00 ± 0.00	0.00^{-}	37.00 ± 0.00	0.00^{-}	6.90 ± 0.57	0.00^{+}
Ring	20.00 ± 0.00	16.00 ± 0.00	0.00^{+}	20.00 ± 0.00	1.00^{o}	20.00 ± 0.00	1.00^{o}	20.00 ± 0.00	1.00^{o}	17.90 ± 0.32	0.00^{+}
Gamma	10.00 ± 0.00	10.00 ± 0.00	1.00^{o}	10.00 ± 0.00	1.00^{o}	10.00 ± 0.00	1.00^{o}	10.00 ± 0.00	1.00^{o}	9.00 ± 0.00	0.00^{+}
Satimage	36.00 ± 0.00	15.20 ± 1.69	0.00^{+}	36.00 ± 0.00	1.00^{o}	36.00 ± 0.00	1.00^{o}	36.00 ± 0.00	1.00^{o}	12.20 ± 0.92	0.00^{+}
Shuttle	3.40 ± 0.52	4.90 ± 0.32	0.00^{-}	*		*		*		6.00 ± 0.00	0.00^{-}
Musk2	138.80 ± 8.23	13.50 ± 0.53	0.00^{+}	*		*		*		#	
Sensorless	6.30 ± 1.16	9.80 ± 2.94	0.00^{-}	*		*		*		#	
MiniBooNE	50.00 ± 0.00	22.80 ± 0.79	0.00^{+}	*		*		*		#	
Winnipeg	167.80 ± 0.63	19.50 ± 1.78	0.00^{+}	*		*		*		#	
Average (NOD)	35.30 (17)	11.94 (17)		22.54 (12)		22.80 (12)		25.11 (12)		14.58 (13)	
CAverage$^{\$}$	21.55	11.13		22.54		22.80		25.11		15.07	
Lose/Win/Tie		2/12/3		5/3/4		5/3/4		6/1/5		3/8/2	

NOD: Number of datasets over which the average is computed (indicated in bracket).

$: Average mean value over 7 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.

Analysis of Results

Classification accuracy results

Table 5.4, Table 5.5 and Table 5.6 show the classification results of CART, kNN and kNN-FMNN classifiers. In all classifiers, the CAverage value of the proposed algorithm CDM-FMFRS is higher than compared algorithms (including FDM-FMFRS) and very near to UNRED.

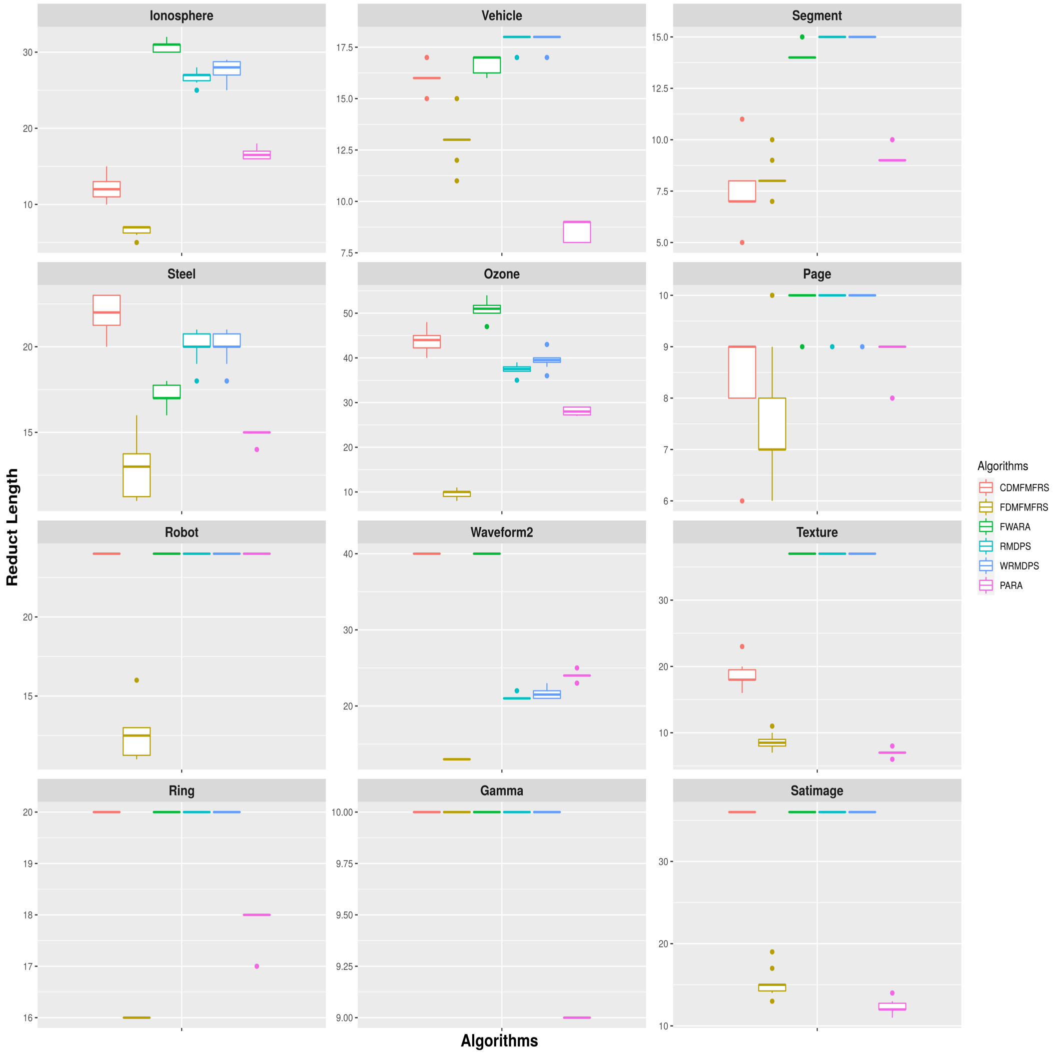
In Table 5.4, considering the overall 83 accuracy results across all the compared algorithms (including FDM-FMFRS) and UNRED in CART classifier, the cumulative lose/win/tie results are 12/3/68. In 68 classification results, the proposed algorithm CDM-FMFRS performed statistically similar to compared algorithms and UNRED. Also, it is observed that wherever CDM-FMFRS performed a little inferior to compared algorithms and UNRED (i.e., 3 results), the differences in average mean are very small. In the remaining 12 results, the proposed algorithm FDM-FMFRS performed significantly better than the compared algorithms, and here also, it is observed that the difference in mean value is small.

Similarly, in other kNN and kNN-FMNN classifiers, as given in Table 5.5 and Table 5.6, majorly all algorithms performed statistically similar to each other. The cumulative lose/win/tie results in kNN classifier is 9/10/64 and in kNN-FMNN is 11/4/68. The further observation analysis details are given below.

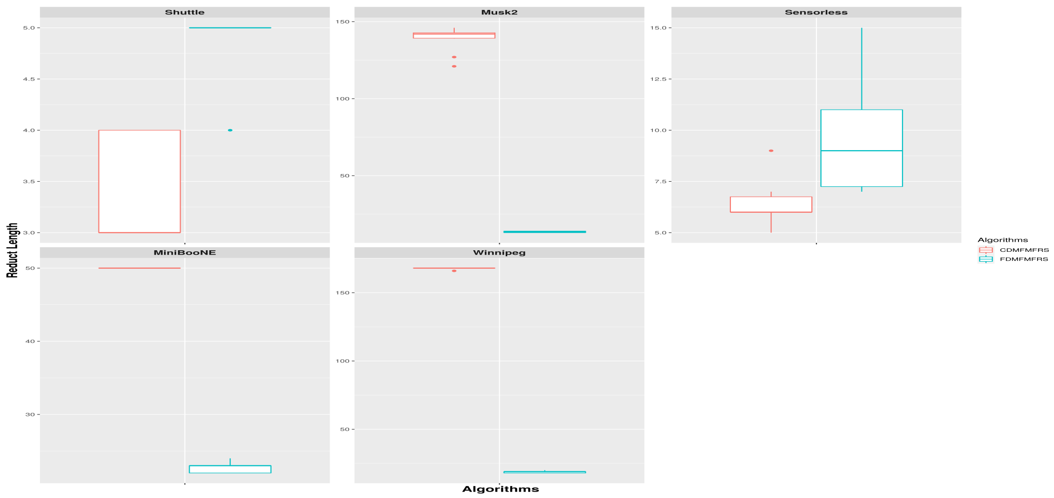
CDM-FMFRS achieved statistically better than RMDPS, WRMDPS and PARA algorithms in Waveform2 dataset in all classifiers, as shown in Fig. 5.1, 5.2 and 5.3. In Shuttle datasets, CDM-FMFRS performed statistically inferior to FDM-FMFRS and UNRED in all classifiers, but differences in mean classification accuracy is very less.

Based on results in Table 5.4 and Fig. 5.1, in Segment, Steel, Robot, MiniBooNE and Winnipeg datasets, CDM-FMFRS incurred statistically better classification results than FDM-FMFRS in CART classifier. Also, It resulted statistically better than RMDPS, PARA and UNRED in Segment dataset. In Shuttle and Sensorless datasets, CDM-FMFRS obtained statistically inferior results than UNRED, but the difference in their results is almost quite low, for example, in Shuttle, CDM-FMFRS is 98.68 and UNRED is 99.96. Moreover, CDM-FMFRS resulted in statistically similar or significant results in most of the datasets in CART classifier.

The following observations are made for the results given in Table 5.5 and Fig. 5.2 using kNN classifier. CDM-FMFRS achieved statistically better results than FDM-FMFRS in Texture, Satimage, Musk2 and MiniBooNE datasets. CDM-FMFRS is statistically inferior to compared algorithms (except PARA) in Texture dataset, but the difference in their average classification accuracies are very less on these datasets. CDM-FMFRS performed statistically



(a) Datasets having reduct length results by all algorithms



(b) Datasets having reduct length results by FDM-FMFRS and CDM-FMFRS

Figure 5.5: Boxplot for Reduct Length Results of Table 5.8

inferior to FDM-FMFRS and UNRED in Sensorless dataset with a high difference in their average classification accuracy. CDM-FMFRS resulted in statistically similar or significant results in most of the datasets in kNN classifier.

In kNN-FMNN classifier, based on Table 5.6 and Fig. 5.3, CDM-FMFRS obtained statistically better than FDM-FMFRS in Segment, Musk2, Sensorless and MiniBooNE datasets. However, CDM-FMFRS performed statistically inferior to FDM-FMFRS in Winnipeg datasets with a minor difference in their average classification accuracy. CDM-FMFRS resulted in statistically similar results in most of the datasets in kNN-FMNN classifier.

Eventually, it can be seen that the idea of computing the approximate reduct by CDM-FMFRS is satisfactory and effective in terms of classification results in the given classifiers. Also, It is further observed that RMDPS, WRMDPS, FWARA and PARA algorithms could not obtain reduct in Shuttle, Musk2, Sensorless, MinibooNE and Winnepeg datasets due to memory overflow (Sign '*') or non-termination even after 24 hours (Sign '#') at given system configuration where CDM-FMFRS can obtain reduct in few seconds.

Computational time results

In terms of computational time given in Table 5.7 and Fig. 5.4, CDM-FMFRS algorithm achieved significantly less computational time than compared algorithms in all datasets and evidently seen that the cumulative lose/win/tie results of compared algorithms are 63/2/1. Against FDM-FMFRS, CDM-FMFRS obtained statistically less computational time in most of the datasets. The proposed method CDM-FMFRS obtained the lowest CAverage value (1.68 seconds) on datasets, whereas compared algorithms and UNRED with CAverage showed a range between 3 and 267 seconds. The average mean value of CDM-FMFRS on 17 datasets is 35.15 seconds, which is much smaller than FDM-FMFRS and compared algorithms. Even the resulting standard deviation of computation time in 10-FCV carried very less variation as compared with compared approaches showing that the methodology is reliable.

Basically, both FDM-FMFRS and CDM-FMFRS algorithms achieved much less computational times than other algorithms. This is due to utility arising from FMNN preprocessing, which makes both algorithms operate in hyperbox space rather than object space where $|HBS| << |U|$ in all datasets. Further, it is observed that CDM-FMFRS achieved better computation time than FDM-FMFRS. This is attributed to using crisp DM in CDM-FMFRS in contrast to fuzzy DM in FDM-FMFRS.

Reduct length results

The results given in Table 5.8 and Fig. 5.4 established that CDM-FMFRS obtained reduct with a statistically significant larger size than FDM-FMFRS on most of the datasets because of adapting crisp DM formulation against fuzzy DM formulation. In FDM-FMFRS, the partial fuzzy membership is calculated based on s-norm computation, and it satisfies the total required s-norm for the entire DM entry with a fewer number of attributes. In FDM-FMFRS, if one attribute is selected, then it contributes some partial membership value to all the entries in fuzzy DM. Whereas, in CDM-FMFRS, if one attribute is selected, then it contributes only to the entries in crisp DM containing that attribute and does no effect on the remaining entries. Hence, it is observed that the average reduct size in FDM-FMFRS is lesser than CDM-FMFRS in most of the datasets. Evidently, the cumulative lose/win/tie results of compared algorithms w.r.t. CDM-FMFRS are 21/27/18.

Moreover, CDM-FMFRS obtained a statistically lesser reduct size than compared algorithms (except FDM-FMFRS) on a few datasets, but the quality of reduct from CDM-FMFRS in terms of average classification accuracies statistically is not compromised.

5.5.3 Role of crisp DM in increased scalability of CDM-FMFRS over FDM-FMFRS

On given datasets, we have seen that in spite of getting higher reduct lengths, CDM-FMFRS methodology has obtained significant gain in computational time over FDM-FMFRS. It is seeing that despite restricting to crisp DM construction, CDM-FMFRS is able to give good quality approximate reduct, which could induce a classification model with comparable or better accuracy. So far, all the experiments are conducted on such datasets over which compared algorithms can be executed for demonstrating the quality of CDM-FMFRS approximate reduct comparatively.

In this section, we demonstrate the improved scalability of CDM-FMFRS over FDM-FMFRS, which is the prime objective for the proposed work. It is also to be noted that all the other FRS reduct approaches are not executable on considered datasets in this experiment owing to memory overflow.

Table 5.2 also gives details of the big numeric datasets (Susy, Swarm Behaviour and Hepmass) used in this experiment. Both datasets Susy, Hepmass and Swarm Behaviour, along with few given datasets from Table 5.2, are considered for experiments. A random sample of Hepmass dataset is considered with 500000 objects.

We have applied CDM-FMFRS and FDM-FMFRS on these datasets, and detailed results are reported in Table 5.9. As the stage for FMNN preprocessing is common to the algorithm, the number of hyperboxes and computational time (in seconds) for FMNN preprocessing is specified only once. Each column in Table 5.9 reports the result size of DM matrix (in MB), DM construction time (in seconds), Reduct Computation Time (in seconds), Reduct Size and Total Time (in seconds) of both CDM-FMFRS and FDM-FMFRS algorithms along with percentage gain of CDM-FMFRS over FDM-FMFRS.

Table 5.9: Experiment Results of CDM-FMFRS and FDM-FMFRS

Datasets	**FMNN**		**DM Size**			**DM Time**		
	NoH	**ToFM**	**CDM-FMFRS**	**FDM-FMFRS**	**Gain (%)**	**CDM-FMFRS**	**FDM-FMFRS**	**Gain (%)**
Shuttle	13	0.33	0.0006	0.0045	87.50	0.0185	0.0201	7.92
Sensorless	26	0.37	0.0136	0.1088	87.50	0.0205	0.0233	12.11
MiniBooNE	2074	144.25	32.82	262.64	87.50	2.74	6.0281	54.59
Winnipeg	3881	246.18	952.27	7618.19	87.50	47.14	82.01	42.52
Susy	13356	7254.8	630.78	5048.32	87.50	82.30	90.42	8.92
Hepmass	26212	2998.8	4741.12	*	*	478.8	*	*
SwarmB	2528	856.95	2742.6	*	*	121.8	*	*

Datsets	**Reduct Time**			**Reduct Size**			**Total Time**		
	CDM-FMFRS	**FDM-FMFRS**	**Gain (%)**	**CDM-FMFRS**	**FDM-FMFRS**	**Gain (%)**	**CDM-FMFRS**	**FDM-FMFRS**	**Gain (%)**
Shuttle	0.0007	0.012	94.65	4	5	20.00	0.3505	0.3658	4.19
Sensorless	0.0013	0.034	96.22	7	7	0.00	0.3586	0.3867	7.26
MiniBooNE	1.04	81.31	98.73	50	22	-127.27	154.12	231.5956	33.45
Winnipeg	32.53	2323.16	98.60	167	20	-735.00	298.08	2651.35	88.76
Susy	4.82	831.01	99.41	18	18	0	7341.12	8175.43	10.20
Hepmass	6.47	*	*	26	*	*	3484.07	*	*
SwarmB	5.16	*	*	50	*	*	983.94	*	*

Note:-'*' represents non-executable due to memory overflow. NoH: Number of Hyperboxes, ToFM: Time for FMNN construction (in seconds), Gain (%): Percentage gain of CDM-FMFRS over FDM-FMFRS, DM: Discernibility Matrix, DM Memory Size (in MegaBytes), DM Time (in Seconds), Reduct Time (in Seconds), Total Time (in Seconds).

Analysis of results

Based on the results in Table 5.9, CDM-FMFRS achieved significant gain in DM memory size as compared with FDM-FMFRS with the same percentage of 87.50% in all datasets. Because in Matlab environment, a real number is represented in 8 bytes, and the logical number is represented in 1 byte. Hence, crisp DM size of CDM-FMFRS is 1/8 of fuzzy DM size of FDM-FMFRS. In other programming environments where the logical value is represented in 1 bit, we would obtain a reduction of 1/64 size of fuzzy DM. Also, CDM-FMFRS obtained a significant gain on time for the construction of DM over FDM-FMFRS with a range of 8-60% on given datasets.

Furthermore, there is a significant percentage gain in reduct computation time on constructed DM in CDM-FMFRS over FDM-FMFRS, more than 90% in all datasets. This is due to Johnson algorithms [144] having lesser computations when applied to crisp DM in comparison to being applied on fuzzy DM. Hence, the total computation time gain of CDM-FMFRS was achieved with a range of 4 to 88.79% in given datasets against FDM-FMFRS algorithm.

In Susy dataset, CDM-FMFRS computed reduct in a significantly lower time of around 5 seconds than 831 seconds in FDM-FMFRS. Additionally, DM construction time for CDM-FMFRS has a slight gain of 8% against FDM-FMFRS, and also a significant reduction in the size of DM in RAM is obtained in CDM-FMFRS. DM size in CDM-FMFRS (630.78 MB) is 1/8 of DM size of FDM-FMFRS (5048.32 MB).

CDM-FMFRS could obtain reduct in Hepmass and SwarmB datasets, whereas FDM-FMFRS failed to do so because of memory overflow. Because the current system employed with 32 GB RAM and hence the requirement of fuzzy DM for FDM-FMFRS for Hepmass dataset in FDM-FMFRS would have been 37.04GB (as the size of crisp DM is 4.63GB size, therefore fuzzy DM size would be 37.04 $((4.63GB) \times 8\ (> 32GB))$ size and which is the reason for FDM-FMFRS is failing the reduct computation as required memory size exceeds available memory of 32 GB. This experiment vividly demonstrates the increased applicability of CDM-FMFRS to much larger numeric datasets and establishes the relevance of CDM-FMFRS over FDM-FMFRS.

Based on these results, one can clearly say that crisp DM formulation significantly reduces the size of DM and reduct computation time. CDM-FMFRS facilitates increased scalability with the disadvantage of a higher length reduct than FDM-FMFRS due to information loss in the crisp DM formulation. Even though we obtain a higher size reduct in some datasets through crisp formulation, the quality of reduct is not compromised as clearly established in obtained Gamma measure showing in Section 5.5.1 and comparable classification model construction in Section 5.5.2. Even, tolerance parameter enriched the quality of reduct. Hence, we recommend CDM-FMFRS as an alternative to FDM-FMFRS in a situation where FDM-FMFRS fails to obtain reduct owing to a memory overflow error.

5.6 Summary

The proposed work (CDM-FMFRS) is an improved mechanism of FDM-FMFRS method to enhance the scalability of reduct computation in hyperbox-space. In CDM-FMFRS, a novel approach for crisp DM formulation in IDS is proposed subject to tolerance criteria for preserving maximal discernible attributes. Hence, the approach achieved significant gain in computation time over FDM-FMFRS and other existing FRS reduct approaches on given benchmark datasets with similar or better classification accuracies over induced different classifiers. Even the space utilization of crisp DM is $\frac{1}{8\times k}$ of space utilization of fuzzy DM. Moreover, CDM-FMFRS approach can handle very large datasets where FDM-FMFRS and other existing state-of-art FRS reduct approaches fail to obtain reduct. In the future, distributed/parallel algorithms for CDM-FMFRS will be investigated for scaling to such voluminous datasets requiring memory beyond the availability in a single system. CDM-FMFRS resulted in higher length reduct due to crisp DM formulation. So, we improve the crisp DM formulation with objective of reducing reduct length in the future work.

Chapter 6

Incremental Feature Subset Selection using Fuzzy Rough Sets with Fuzzy Min-Max Neural Network Preprocessing

Chapter 4 and Chapter 5 provide FRS-based feature subset selection approaches. These approaches are restricted to batch processing; the entire data and its underlying structure are provided prior to training at once. However, they are not designed for dealing with dynamic datasets. When a new sample data arrives, these approaches have to recompute and reconstruct the model from scratch to learn new data and compute a reduct. Hence, these FRS algorithms suffer a lack of model adaptability, i.e., not continuously integrating new information into existing models on continually succeeding new information/data. One solution is to implement the incremental technique to handle dynamic datasets and update a reduct dynamically on data arrival. The challenge of the incremental strategy is to retain the previously acquired knowledge while acquiring new information. This chapter focuses on an incremental FRS-based feature selection algorithm using FMNN preprocessing.

Section 6.1 briefly introduces the literature survey of incremental FRS approaches and their limitations. Section 6.2 presents the motivation of the proposed algorithm. Section 6.3 briefly describes the functioning of the proposed incremental algorithm IvFMFRS. Section 6.4 describes the complexity analysis of proposed algorithm IvFMFRS. Section 6.5 reports a series of experiments and comparative analysis of IvFMFRS with state-of-the-art incremental approaches.

6.1 Existing Approaches

Based on the literature reviewed in Chapter 4, many of the existing FRS approaches are implemented in a classical batch setting, which can handle all data at once prior to training, and training can rely on the assumption that the data and its underlying structure are static [30, 119]. These approaches suffer from continuous model adaptation, i.e., not continuously integrating new information into existing models on constantly (subsequently) arriving new information/data. Hence, this results in the recomputation and reconstruction of new models from scratch, which is repeatedly a time-consuming task.

The incremental learning process is a machine learning paradigm that extends and learns the existing model's knowledge whenever new examples emerge without losing previous information/data [30]. The primary objective for incremental learning is to update or learn a continual basis of knowledge on constant arriving at new data. Incremental learning drives the limit of current learning systems over time with data [30]. Thus, this property becomes essential for discovering knowledge and an important facet of human intelligence.

In the last decades, several researchers have explored how to process dynamic data through incremental learning methodologies that minimize the complexities of processing and storage. This idea has prompted several researchers to investigate the incremental perspective to feature selection in the framework of RST for categorical decision systems. These ideas have been investigated in various scenarios, such as the variation of feature set (adding and deleting features) and the sample set (adding and deleting objects), respectively. For incrementally adding and deleting features, some incremental reduct computation algorithms are introduced based on information entropy [115], discernibility matrix [133], knowledge granularity [46, 82] and positive domain [98]. For incrementally adding and deleting objects, there are some incremental algorithms based on information entropy [16, 96, 114], discernibility matrix [59, 117], knowledge granularity [47], positive domain [97], bijective soft sets [75] and representative instances [118].

There have been a few studies on FRS based incremental feature selection algorithms. As our work is based on an incremental approach under object space variation, the associated literature is briefly described here.

In 2017, Yang et al. [120] proposed two incremental algorithms for feature selection based on FRS for dealing with dynamic datasets. These two incremental algorithms are designed primarily upon the arrival of one sample and multiple samples over time. On arrival of the sample subset (one or multiple), the approach updates the relative discernibility relation for each conditional attribute. Then, an incremental process is to update the current reduct by adding new attributes and deleting redundant attributes based on updated discernibility relations.

Again, in 2018, Yang et al. [119] proposed two incremental feature selection algorithms (IV-FS-FRS(1) and IV-FS-FRS(2)) based on FRS, which is an extension of [120] on dynamic datasets. These approaches provide a way to add and delete attributes from the current reduct based on updated relative discernibility relations on sample subsets arrival. The authors designed two algorithms to update the current reduct with each incoming subsets arrival. One (IV-FS-FRS(1)) is to incrementally update only relative discernibility relation on subsequent arrival of sample subsets but only perform feature selection when no further sample subset is left. Another (IV-FS-FRS(2)) is to update the relative DM incrementally with an incoming sample and then update the corresponding current reduct with adding and deleting attributes through an updated discernibility matrix. We are using IV-FS-FRS(2) algorithm for our comparison with proposed algorithm.

The aforementioned incremental algorithms perform discernibility matrix-based computation [119, 120] and their corresponding feature selection. However, these algorithms require a large amount of memory space which is sometimes intractable for large decision systems.

In 2020, Zhang et al. [135] proposed an incremental feature selection algorithm (AIFWAR) based on FRS using information entropy on new incoming subsets. Information entropy doesn't require much memory compared to relative discernibility matrix construction. In [135], the author aims first to select representative instances from the arriving sample subset using FRS concept, and then an incremental mechanism of the information entropy is measured using representative instances. Then, a corresponding incremental feature selection approach is developed by using information entropy. Finally, a wrapper procedure is applied to the resultant feature subset to select the best features that achieve maximum accuracy by inducing a classification model.

In 2020, Peng et al. [73] introduced a positive region-based incremental feature selection (PIAR) using FRS concepts. The author's idea is to select key instance set containing representative instances on arriving sample subsets. These instances consist of all instances that do not reach the maximum positive region values. Then based on key instances, the incremental mechanism of updating current reduct with adding attributes with the current reduct and eliminating redundant attributes using dependency degree measure.

6.2 Motivation

The aforementioned incremental FRS algorithms [119, 120] require object-based computation that impacts an increase in space and computation overhead. Sometimes it is impossible to load discernibility matrix entries on memory for new information/data. Even selecting the representative instances, given in [73, 135], from incoming instances also requires additional computational time. They require the generation of fuzzy similarity matrices beforehand to select representative instances. An increase in object space would adversely impact computational overhead on these approaches.

Based on the results of FDM-FMFRS in Chapter 4, we have established the utility of the granular computing aspect for reduct computation in a batch environment. And in the entire literature review of incremental mechanism, we have not noticed any such utilization of granular computing aspect in incremental reduct computation, which can significantly reduce time and space computation. This motivates us to investigate the incremental perspective of FRS approach on reducing the space complexity that can enhance the scalability of incremental FRS reduct computation.

In this chapter, such an intuitive idea is introduced for a solution to incremental FRS feature subset selection by using the concept of hyperbox utilizing FMNN as a preprocessor.

6.3 Proposed Approach

This section describes the proposed FRS incremental algorithm IvFMFRS (Iv:Incremental version, FM: Fuzzy Min-Max Neural Network, FRS: Fuzzy Rough Set) to compute an approximate reduct utilizing FMNN learning model. The proposed algorithm has extended the FDM-FMFRS, described in Chapter 4, to an incremental perspective for computing a reduct for the real-valued dynamic decision system, where samples data are arriving sequentially.

6.3.1 Incremental Environment Description and Notation

This section describes the incremental environment and the used symbols/notations in algorithms. The description of function and notation inside the Algorithm [6.1], Algorithm [6.2] and Algorithm [6.3] are mentioned in Table 6.1. Also, we present a flowchart of IvFMFRS algorithm for better understandability and as depicted in Fig. 6.1.

Here, we assume that the data is presented in sample subsets $(U_1, U_2, U_3, \dots)$ that arriving sequentially. So, in each iteration, a new sample subset is provided to an algorithm to perform incrementally.

Every incremental algorithm starts with the corresponding base algorithm. For our case, we used FDM-FMFRS, described in Chapter 4, as the base algorithm. Initially, we compute a set of hyperboxes HBS_1, fuzzy DM M_1 and base reduct R_1 through FDM-FMFRS for a sample U_1 to further incremental computation. For the next sample subset arrival U_2, we apply our incremental IvFMFRS with given HBS_1, M_1 and R_1 to incrementally compute

Table 6.1: Description of Function Name and Notation in Algorithms

Notation	Meaning
FM	Represents FMNN learning model.
$FM.Belongs(x)$	Checks absolute membership value (Eqn. (2.2)) of x on any existing hyperboxes of same class.
$Next$	Breaks the current iteration and continues the next iteration in the loop.
$FM.HMemb(x)$	Finds the highest membership value correspond to x with existing hyperbox of same class label.
$FM.Exp(H, x)$	Checks expansion of H to include x is possible or not using expansion criterion Eqn. (2.6).
$Remove(HBS, H)$	Removes H from set HBS.
$FM.Expand(H, x)$	Expands the hyperbox H to include x using Eqn. (2.7) and Eqn. (2.8).
$Insert(HBS, H)$	Inserts newly H into set of hyperbox HBS.
$Update(HBS, H)$	Updates the expanded hyperbox H in the set HBS.
$FM.Create(x)$	Creates a new point hyperbox to include x.
$A \cup B$	Merge of A and B.

HBS_2, M_2 and reduct R_2. Similarly, the algorithm is repeated for subsequent samples.

Basically, for arriving sample U_{i+1}, IvFMFRS is initialized with HBS_i, M_i and R_i as inputs to compute HBS_{i+1}, M_{i+1} and R_{i+1} as outputs.

The proposed incremental step of R_{i+1} computation involves the following steps:

1. Updating HBS_i with incremental training on U_{i+1} as HBS_{i+1}.

2. Updating M_i with HBS_{i+1} as M_{i+1}.

3. Updating R_i on M_{i+1} for getting R_{i+1}.

6.3.2 Updating fuzzy hyperboxes through FMNN learning model

This section shows the updation of existing hyperboxes HBS_i by training FMNN with a new batch sample U_{i+1}. The procedures for implementing FMNN is similar to the proposed work kNN-FMNN as described in Chapter 3. Algorithm [6.1] performs updation of hyperboxes. For incubation of an input pattern x in hyperbox space, if x gives an absolute membership value with any existing hyperbox representing the same class using Eqn. (2.2), then no modification on hyperbox takes place. If x is outside the hyperbox, then a hyperbox H corresponding to the highest membership value is selected to verify whether it can be expanded or not using expansion criterion Eqn. (2.6). If yes, then hyperbox H is expanded to accommodate input x by adjusting their min and max points of H using Eqn. (2.7) and Eqn. (2.8). If not, then a

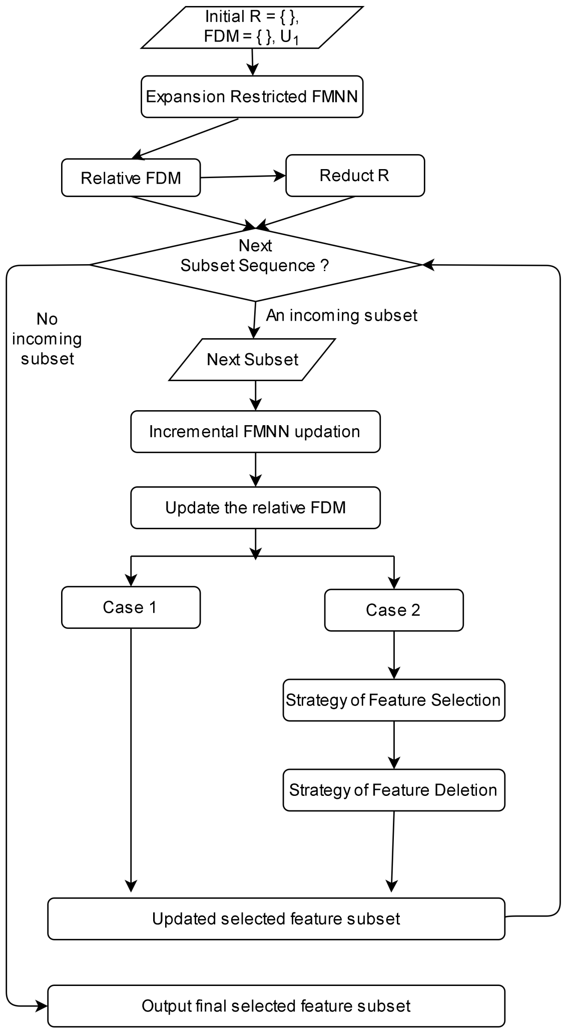


Figure 6.1: Flow Chart of IvFMFRS

hyperbox with the next highest membership value is chosen for expansion to include pattern x. This process continues until any hyperbox can include the input pattern x. If none of the hyperboxes is met expansion criteria, then a new point hyperbox is created to incorporate the input pattern x.

After completion of training with all input patterns in U_{i+1}, hyperboxes are divided into three categories:

1. Hyperboxes exist in HBS_i but are not modified.

2. Hyperboxes exist in HBS_i but are modified as part of the expansion process. These hyperboxes are removed from HBS_i and saved in set HBS^{mod}.

3. Newly created hyperboxes that represent input patterns are saved in set HBS^{new}. These hyperboxes may be updated as part of the expansion process.

Hence, the final hyperbox space $HBS_{i+1} = HBS_i \cup HBS^{mod} \cup HBS^{new}$.

Algorithm 6.1: Updating Fuzzy Hyperboxes through FMNN

Input : U_{i+1}, HBS_i
Output: $HBS_{i+1},\ HBS_i,\ HBS^{mod},\ HBS^{new}$

Initialize, $HBS^{mod} = \emptyset, HBS^{new} = \emptyset$;
// Let FM represents FMNN model comprises $HBS_i \cup HBS^{mod} \cup HBS^{new}$
for *every* x *in* U_{i+1} **do**
    if $FM.Belongs(x) == True$ **then**
        Next;
    end
    HS = FM.HMemb(x);
    Flag = 0;
    if $HS \neq \emptyset$ **then**
        for *every* H *in* HS **do**
            if $FM.Exp(H, x) == True$ **then**
                if $H \in HBS_i$ **then**
                    Remove(HBS_i, H); FM.Expand(H, x);
                    Insert(HBS^{mod}, H);
                else if $H \in HBS^{new}$ **then**
                    FM.Expand(H, x); Update(HBS^{new}, H);
                else if $H \in HBS^{mod}$ **then**
                    FM.Expand(H, x); Update(HBS^{mod}, H);
                    Flag = 1;
                    Break;
                end
            end
            if *Flag == 0* **then**
                H^{new} = FM.Create(x);
                Insert(HBS^{new}, H^{new});
            end
        end
    else
        H^{new} = FM.Create(x);
        Insert(HBS^{new}, H^{new});
    end
end
$HBS_{i+1} = HBS_i \cup H^{mod} \cup H^{new}$;
return $HBS_{i+1}, HBS_i, HBS^{mod}, HBS^{new}$

6.3.3 Updating Fuzzy Discernibility Matrix

This section focuses on updating fuzzy DM (M_i), and the approach depends on the categories of hyperboxes presented in the previous section. Algorithm [6.2] performs an update of fuzzy DM. Hyperboxes in the first category HBS_i are unmodified; hence they don't contribute to changes into any existing fuzzy DM entries. In an ideal scenario, we have the hyperboxes in HBS_i that are representative of all-new patterns in U_{i+1}. In that case, no modification of hyperboxes structure occurs which mean $HBS_{i+1} = HBS_i$. Hence, there is no update of fuzzy DM, and the reduct remains unchanged. So, R_i becomes R_{i+1} and the algorithm immediately returns reduct. The chance for this ideal scenario increases as more training data arrives.

Whereas in the second category, modified hyperboxes HBS^{mod} change the learning model by adjusting their V (min point) and W (max point) values so that their respective entries in M_i need to be modified.

For the third category, the hyperboxes in HBS^{new} are the new objects augmented to current IDS. For their fuzzy DM entries, corresponding new entries are added to the existing fuzzy DM. These hyperboxes are compared with different class hyperboxes of HBS_{i+1}.

Modified entries of fuzzy DM resulting from second and third category hyperboxes are removed from M_i and placed in a new collection M^{new} representing either new or updated entries of current fuzzy DM.

The final fuzzy DM is $M_{i+1} = M_i \cup M^{new}$.

Algorithm 6.2: Updating Fuzzy Discernibility Matrix

Input : HBS_i, HBS^{mod}, HBS^{new}, M_i
Output: M_{i+1}, M^{new}

Initialize, $M^{new} = \emptyset$;
for *each* $H^m \in HBS^{mod}$ **do**
  for *each* $H \in (HBS_i \cup HBS^{mod})$ **do**
    if $d(H^m) \neq d(H)$ **then**
      `// Update` $M_i(H^m, H)$ `using Eqn. (4.17)`
      $M^{new} = M^{new} \cup \{M_i(H^m, H)\}$;
    end
  end
end
`// Remove all updated entries from` M_i
for *each* $H^n \in HBS^{new}$ **do**
  for *each* $H \in \{HBS_i \cup HBS^{mod} \cup HBS^{new}\}$ **do**
    if $d(H^n) \neq d(H)$ **then**
      `// Compute` $M(H^n, H)$ `using Eqn. (4.17)`
      $M^{new} = M^{new} \cup \{M(H^n, H)\}$;
    end
  end
end
$M_{i+1} = M_i \cup M^{new}$;
return M_{i+1}, M^{new}

6.3.3.1 Remark

There is an advantage of the granularity concept in our proposed algorithm over any object-based incremental learning approach. In existing object-based incremental approaches [73, 119, 120, 130, 135], for every new object arrival, new fuzzy DM entries must be created which is a time-consuming task. But in the proposed approach, for all the new training input patterns that have obtained absolute membership into any of the existing hyperboxes, no fuzzy DM update is required reducing frequent alterations of fuzzy DM. This significantly diminishes the computational times and space requirements of the proposed approach.

6.3.4 Incremental Computation of Reduct

Algorithm [6.3] performs an update of current reduct R_i to become R_{i+1}. After updating fuzzy DM, as discussed in the above section, the incremental process for updating current reduct R_i is performed by using two case strategies, as summarized below:

$Case$ 1: $SAT_{M^{new}}(R_i) == SAT_{M^{new}}(C^n)$;

$Case$ 2: $SAT_{M^{new}}(R_i) \neq SAT_{M^{new}}(C^n)$;

If the $Case$ 1 holds, the current reduct R_i is satisfied the newly added or modified entries in M^{new} and R_i is already satisfied unmodified entries (old records) of M_i. Hence, the updation of the current reduct is not required, and existing R_i becomes reduct of M_{i+1}. So, current reduct R_{i+1} is R_i for sample U_{i+1}.

If the $Case$ 2 holds, the current reduct R_i doesn't satisfy some of the newly added entries in M^{new} leading to requirement for update of R_i for becoming reduct. The process of computing R_{i+1} is started with initializing R_{i+1} to R_i. The remaining computations are done in two phases.

In the first phase, the additional attributes are added ($\forall c \in C^n - R_{i+1}$) into R_{i+1} using SFS strategy apply only on M^{new}. Here in each iteration, SAT measure is computed with a different attribute that is not already included in R_{i+1}, given in Eqn. (4.16) for $(R_{i+1} \cup \{c\})$ $\forall c \in C^n - R_{i+1}$. Then, the attribute having a maximum SAT measure is included in R_{i+1}. This strategy for attribute selection is repeated till $SAT_{M^{new}}(R_{i+1}) = SAT_{M^{new}}(C^n)$. The SFS strategy of reduct updation is only restricted to M^{new} as R_{i+1} already satisfies unmodified entries in M_i.

The modified R_{i+1} is a super reduct for $M_{i+1}(= M_i \cup M^{new})$ and can contain the redundant attributes. Hence, in the second phase, SBE strategy in [31], which is an efficient third order complexity approach, is followed on M_{i+1} to remove redundant attributes in R_{i+1}. Here for each attribute 'c' in R_{i+1}, it is checked whether omission of the attribute 'c' affects SAT measure. It is verified whether $SAT_{M_{i+1}}(R_{i+1} - \{c\})$ is one or not. If one, then the attribute 'c' is redundant and hence removed from R_{i+1}. Otherwise, the attribute 'c' is indispensable and retained in R_{i+1}.

Finally, the current R_{i+1} is the final reduct for samples $\bigcup_{j=1}^{i+1} U_j$.

Algorithm 6.3: Incremental Way to Compute Reduct

Input : R_i, M^{new}, M_{i+1}
Output: R_{i+1}

if $SAT_{M^{new}}(R_i) == SAT_{M^{new}}(C^n)$ **then**
    $R_{i+1} = R_i$;
else
    $R_{i+1} = R_i$;
    while $SAT_{M^{new}}(R_{i+1}) \neq SAT_{M^{new}}(C^n)$ **do**
        For each $c \in C^n - R_{i+1}$, Compute $SAT_{M^{new}}(R_{i+1} \cup \{c\})$;
        Select feature $c_o \in C^n - R_{i+1}$, satisfying
        $SAT_{M^{new}}(R_{i+1} \cup \{c_o\}) = max_{c \in C^n - R_{i+1}} SAT_{M^{new}}(R_{i+1} \cup \{c\})$;
        $R_{i+1} = R_{i+1} \cup \{c_o\}$;
    end
    // Compute $SAT_{M_{i+1}}(C^n)$
    for *each* $c \in C^n - R_{i+1}$ **do**
        // Compute $SAT_{M_{i+1}}(R_{i+1} - \{c\})$
        if $SAT_{M_{i+1}}(C^n) == SAT_{M_{i+1}}(R_{i+1} - \{c\})$ **then**
            $R_{i+1} = R_{i+1} - \{c\}$;
        end
    end
end
return R_{i+1}

6.4 Complexity Analysis of IvFMFRS Algorithm

This section shows the time and space complexity analysis of the proposed algorithm IvFMFRS. The following variables are used in the complexity analysis of IvFMFRS.

- $|U_{i+1}|$: the number of objects in U_{i+1}.
- $|HBS_i|$: the number of all hyperboxes in HBS_i based on U_i.
- $|HBS_{i+1}|$: the number of all hyperboxes in HBS_{i+1} based on U_{i+1}.
- $|HBS^{mod}|$: the number of modified hyperboxes in HBS^{mod}.
- $|HBS^{new}|$: the number of newly created hyperboxes in HBS^{new}.
- $|C^n|$: the number of numeric conditional attribute.

- $|M^{new}|$: Size of discernibility matrix M^{new}.
- $|M_{i+1}|$: Size of discernibility matrix M_{i+1}.
- $|R_i|$: Current reduct based on U_i.
- $|R_{i+1}|$: Updated reduct based on U_{i+1}.

Table 6.2 shows the time complexity of the proposed algorithm IvFMFRS for one iteration from U_i to U_{i+1}. In Table 6.2, Algorithm [6.1] with steps 2 to 32 performs updation of IDS using FMNN, with time complexity of $O(|U_{i+1}|*|HBS_{i+1}|*|C^n|)$. In Table 6.2, Algorithm 6.2 with steps 2 to 16 incrementally computes fuzzy DM (M_i) based on IDS from Algorithm 6.1 with time complexity of $O(|HS^{mod} \cup HBS^{new}| * |HBS_{i+1}| * |C^n|)$.

In Table 6.2, Algorithm [6.3] with steps 4-10 performs SFS computation for adding features into the current reduct, with time complexity of $O(|M^{new}| * |C^n - R_i|^2) = O(|HBS^{mod} \cup HBS^{new}|^2 * |C^n - R_i|^2)$. Algorithm [6.3], Steps 11-15 performs the strategies of deleting redundant features in current reduct, with third order time complexity of $O(|M_{i+1}|*|R_{i+1}|) = O(|HBS_{i+1}|^2 * |R_{i+1}|)$.

So, the total time complexity of the proposed algorithm IvFMFRS is: $O(|U_{i+1}|*|HBS_{i+1}|*|C^n|)+O(|HBS^{mod} \cup HBS^{new}|*|HBS_i|*|C^n|)+O(|M^{new}|*|C^n - R_i|^2)+O(|M_{i+1}|*|R_{i+1}|)$.

The space requirement of IvFMFRS in one iteration is for three sources: The decision system U_{i+1} is required for updating IDS with a space complexity of $O(|U_{i+1}|*|C^n|)$. Second, IDS-based fuzzy DM (M_i) is updated with a requirement of space complexity $O(|HBS_{i+1}| * |C^n|)$. Finally, fuzzy DM is required for updating the current reduct having a space complexity $O(|M_{i+1}| * |C^n|) = O(|HBS_{i+1}|^2|C^n|)$.

Thus, the space complexity of IvFMFRS algorithm is $O(|U_{i+1}| * |C^n|) + O(|HBS_{i+1}|^2 * |C^n|)$.

Table 6.2: Time Complexity Analysis of FDM-FMFRS Algorithm

Algorithm (phase)	Steps in Algorithm	Time complexity
Algorithm 6.1	2-32. Updation of IDS using FMNN	$O(\|U_{i+1}\| * \|HBS_{i+1}\| * \|C^n\|)$
Algorithm 6.2	2-16. Updation of fuzzy DM based on IDS	$O(\|HS^{mod} \cup HBS^{new}\| * \|HBS_{i+1}\| * \|C^n\|)$
Algorithm 6.3	4-10. Adding features into current reduct	$O(\|M^{new}\| * \|C^n - R_i\|^2) = O(\|HBS^{mod} \cup HBS^{new}\|^2 * \|C^n - R_i\|^2)$
	11-15. Removing redundant features from current reduct	$O(\|M_{i+1}\| * \|R_{i+1}\|) = O(\|HBS_{i+1}\|^2 * \|R_{i+1}\|)$

6.5 Experiments

This section evaluates the experimental performance of the proposed incremental algorithm IvFMFRS. The comparative analysis of proposed incremental algorithm is conducted with the recent (published in 2018-20) incremental FRS reduct approaches namely IV-FS-FRS [119], AIFWAR [135] and PIAR [73].

6.5.1 Environment and Objectives of Experimentation

Table 6.3: Benchmark Datasets

Dataset	Attributes	Objects	Class
Ionosphere	32	351	2
Vehicle	18	846	4
Segment	16	2310	2
Steel	27	1941	7
Ozone Layer	72	1848	2
Page	10	5472	5
Robot	24	5456	4
Waveform2	40	5000	3
Texture	40	5500	11
Gamma	10	19020	2
Satimage	36	6435	6
Ring	20	7400	2
Musk2	166	6598	2
Shuttle	9	57999	7
Sensorless	48	58509	11
MiniBooNE	50	129596	2
Winnipeg	174	325834	7

Seventeen benchmark datasets of different sizes were collected from the UCI machine learning repository [21] for experimental evaluation, as outlined in Table 6.3. The hardware environment of the system applied for experiments is CPU: Intel(R) i7-8500, Clock Speed: 3.40GHz $\times$ 6, RAM: 32 GB DDR4, OS: Ubuntu 18.04 LTS 64 bit and Software: Matlab R2017a. The proposed algorithm is implemented in the Matlab environment. For IvFMFRS, the Lukasiewicz t-conorm ($S(x, y) = min\{x + y, 1\}$) for Eqn. (4.16) and fuzzy standard negation ($Neg(x) = 1 - x$) for Eqn. (4.17) are employed.

We selected the value of the sensitive parameter *gamma* (γ) to 4 as recommended from the original FMNN paper [102]. Also, we have chosen the *theta* (θ) parameter to 0.3 based on experimental results obtained for base algorithm FDM-FMFRS for restricting hyperboxes size in FMNN learning model [102]. Moreover, the compared algorithms (IV-FS-FRS, AIFWAR and PIAR) follow their fuzzy model of t-norm, t-conorm and fuzzy similarity relations for computing as given in the respective publications and experiments are conducted in the same environment stated above. The performance of IvFMFRS is examined through a comparative evaluation with respect to the following objectives:

1. Evaluate quality of approximate reduct through Gamma measure.
2. Comparative analysis of proposed approach in construction of different classifiers through ten-fold cross-validation (10-FCV).
3. Comparative analysis of incremental reduct algorithms.

6.5.2 Evaluating Quality of Reduct Computed through Gamma Measure

Reduct computation in IvFMFRS is based on a discernibility matrix construction in the hyperbox space. Since fuzzy DM on IDS is an approximation of fuzzy DM on objects, theoretically, it results in an approximate reduct. Hence, some information loss is also present naturally.

The details of Gamma measure are precisely the same as followed in Chapter 4 on page number 61.

Table 6.4 contains the resulting gamma value and reduct length by applying the proposed algorithm as well as the compared algorithms on the entire dataset. We randomly divided the entire dataset into ten equal subsets from an incremental perspective. Each subset sequentially updates the incremental models for reduct computation. The last subset outcome is the final reduct. Also, Table 6.4 represents the gamma measure obtained from the unreduced decision system (mention as 'UNRED' in Table 6.4) to validate the relevance of resulted reducts through checking whether the obtained reduct is satisfying or reaching near to (UNRED) gamma measure or not.

Table 6.4 reports the gamma value for only eleven datasets out of seventeen benchmark datasets due to exceeding the memory limit while processing the GKFRS. And, out of eleven datasets, IV-FS-FRS could compute reduct in only seven datasets.

Analysis of Results

In Table 6.4, it is observed that IvFMFRS achieved an equal gamma measure as obtained by "UNRED" satisfying the required reduct property fully in Ionosphere, Steel, Vehicle, Page, Texture, Waveform2 and Ring datasets. In the remaining datasets, IvFMFRS indeed achieved almost near to expected gamma measure w.r.t the entire dataset gamma value.

Overall, it can be seen that the approximate reduct from IvFMFRS is not resulting in any significant loss in the quality of reduct. Also, it can be observed that the size of reduct for

Table 6.4: Relevance of IvFMFRS reduct through Gamma measure

Datsets	Gamma Meausre				
	UNRED	**IvFMFRS**	**IV-FS-FRS**	**AIFWAR**	**PIAR**
Ionosphere	0.99	0.99	0.99	0.99	0.99
Segment	0.98	0.90	0.14	0.97	0.97
Steel	0.99	0.99	0.80	0.94	0.99
Vehicle	0.99	0.99	0.19	0.99	0.99
Ozone	1	0.99	0.99	1	1
Page	0.87	0.87	0.04	0.87	0.87
Texture	0.99	0.99	*	0.99	0.99
Waveform2	1	1	*	1	1
Robot	0.97	0.91	0.97	0.38	0.97
Satimage	0.99	0.98	*	0.99	0.97
Ring	1	1	*	0.98	1
Datsets	**Reduct Length**				
	UNRED	**IvFMFRS**	**IV-FS-FRS**	**AIFWAR**	**PIAR**
Ionosphere	32	7	9	8	17
Segment	16	8	1	9	9
Steel	27	12	5	6	15
Vehicle	18	12	2	15	10
Ozone	72	10	22	21	28
Page	10	9	2	9	9
Texture	41	13	*	14	7
Waveform2	40	14	*	39	26
Robot	24	14	24	4	24
Satimage	36	15	*	36	13
Ring	20	16	*	7	18

Notes: * represents non-executable due to memory overflow.

IvFMFRS is lesser than AIFWAR and PIAR algorithms in most of the datasets. Moreover, IV-FS-FRS achieved less reduct size than IvFMFRS in some datasets, but their corresponding gamma value is significantly less than our proposed approach in those instances. Even, in Texture, Waveform2, Satimage and Ring datasets, IvFMFRS could compute reduct whereas IV-FS-FRS could not. Hence, empirically, we have established that IvFMFRS results in quality reduct with the same or very similar gamma measure as that of UNRED.

Section 6.5.3 explores the relevance of obtained approximate reduct of IvFMFRS in achieving the construction of the classification learning model, which is the primary objective of the feature subset selection. Moreover, the comparative analysis with reduct length and computational time will be elaborated as part of Section 6.5.3 in tenfold cross-validation.

6.5.3 The Relevance of IvFMFRS Algorithm in Construction of classifiers

This section contains the comparative experiments conducted among algorithms for reduct computation, i.e., IvFMFRS, IV-FS-FRS [119], AIFWAR [135] and PIAR [73] algorithms. The relevance of reduct in inducing a classification model is studied through ten-fold cross-validation (10-FCV) experiments. In each iteration, one fold is preserved for the testing data, and the remaining nine folds are used for training data. For incremental algorithms, we randomly divided the training dataset into ten equal subsets. Each subset sequentially updates the incremental learning model for reduct computation. The last subset outcome of an algorithm is the final reduct for each fold. A reduct algorithm is applied to the training data. So, based on the reduct that is obtained, the classification model is constructed for comparison. The classification accuracy of the resulting model is evaluated based on the test data.

Two different classifier models are used, namely CART and kNN with default options, and for kNN experiments, k is taken as 3, and our proposed kNN-FMNN classifier (Chapter 3) is also employed for inducing classification model. To examine the relevance of reducts, we have also constructed the classification model with an unreduced dataset (mentioned as 'UNRED' in the given Tables) for comparison.

Table 6.5, Table 6.6 and Table 6.7 presents the results of the 10-FCV experiment for classification accuracies with CART, kNN, and kNN-FMNN respectively. Similarly, Table 6.8 and Table 6.9 illustrates the reduct length and computational time of the algorithms. Fig. 6.2, Fig. 6.3, Fig. 6.4, Fig. 6.5 and Fig. 6.6 depict the box-plot representation of Table 6.5, Table 6.6, Table 6.7, Table 6.8 and Table 6.9 respectively.

The detailed student's paired t-test analysis and how the values are represented in Tables 6.5, 6.6, 6.7, 6.8 and 6.9 are precisely the same as followed in Chapter 4 on page number 62.

The last three lines in each Table6.5, 6.6, 6.7, 6.8 and 6.9 correspond to **Average (NOD)**, **CAverage**, and **Lose/Win/Tie**. It can be observed that the datasets over which an algorithm is executing vary from one to another. Hence, the average of individual mean values is reported in two forms. Average (NOD) corresponds to the average value obtained by an algorithm on datasets where it could be evaluated along with reporting the number of datasets (NOD) involved in brackets. CAverage value depicts the average of the individual mean obtained by restricting to only those datasets in which all algorithms could be evaluated. For the comparative analysis, CAverage plays an important role. The last line indicates the count of the number of statistically loss('-'), better('+'), and equivalent('o') for each algorithm in comparison with the proposed IvFMFRS.

Note: The '*' sign in Tables 6.5, 6.6, 6.7, 6.8 and 6.9 and 5.8 shows the corresponding algorithm is intractable to a particular dataset to compute the reduct due to insufficient memory. And, '#' sign represents the scenario of non-termination of the code even after several hours of computation.

In Figures 6.2, 6.3, 6.4, 6.5 and 6.6, the range of Y-axis varies based on obtained results in each dataset. For large datasets, as results are available only for IvFMFRS, AIFWAR and PIAR algorithms, Figures are respectively given in Figure (b) part.

Table 6.5: Classification Accuracies Results (%) with CART in 10-FCV

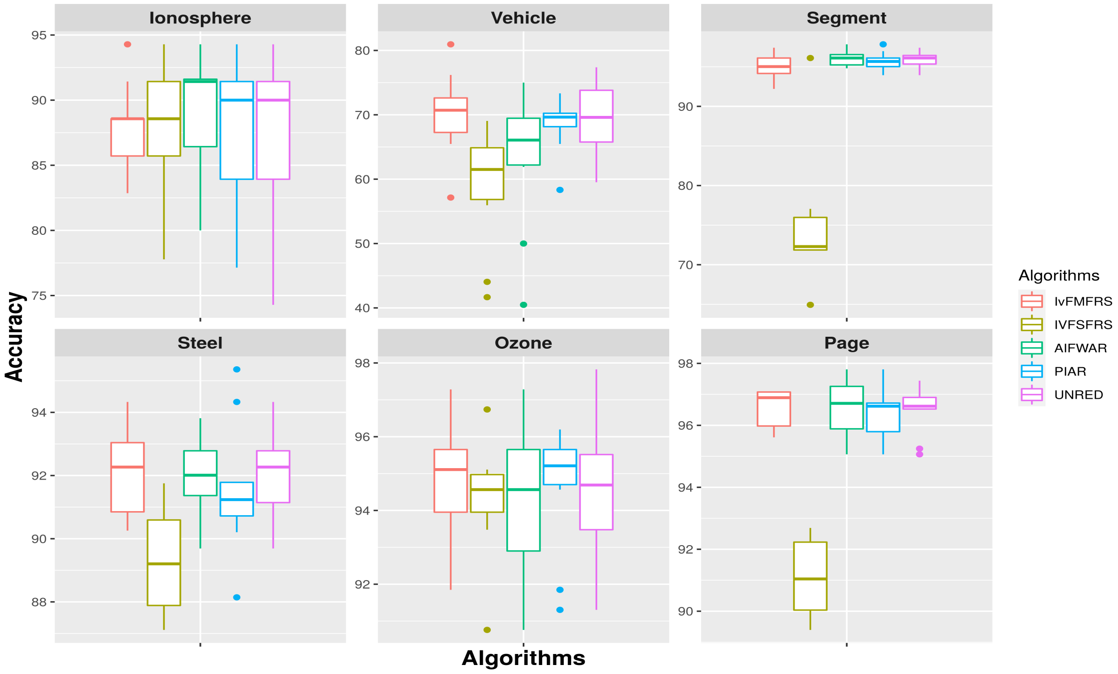
Datasets	**IvFMFRS**	**IV-FS-FRS**		**AIFWAR**		**PIAR**		**UNRED**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	87.76 ± 3.51	87.78 ± 5.59	0.99^o	88.88 ± 5.31	0.58^o	87.48 ± 5.71	0.89^o	87.48 ± 6.59	0.91^o
Vechile	70.21 ± 6.42	59.09 ± 9.47	0.01^-	63.31 ± 10.57	0.09^o	68.52 ± 4.14	0.49^o	69.63 ± 5.79	0.83^o
Segment	95.19 ± 1.63	74.89 ± 8.16	0.00^-	96.02 ± 1.00	0.19^o	95.67 ± 1.22	0.47^o	95.84 ± 1.08	0.31^o
Steel	92.17 ± 1.47	89.28 ± 1.70	0.00^-	92.01 ± 1.15	0.79^o	91.55 ± 2.05	0.45^o	92.22 ± 1.50	0.94^o
Ozone	95.30 ± 1.43	95.46 ± 1.45	0.80^o	95.08 ± 1.32	0.73^o	94.69 ± 1.72	0.77^o	95.41 ± 1.64	0.87^o
Page	96.56 ± 0.64	91.12 ± 1.27	0.00^-	96.56 ± 0.97	1.00^o	96.38 ± 0.85	0.59^o	96.49 ± 0.76	0.82^o
Robot	98.61 ± 0.60	99.50 ± 0.27	0.00^+	92.56 ± 1.17	0.00^-	99.47 ± 0.31	0.00^+	99.47 ± 0.31	0.00^+
Waveform2	74.34 ± 2.46	*		70.38 ± 3.19	0.01^-	71.62 ± 2.24	0.02^-	74.48 ± 1.29	0.88^o
Texture	91.09 ± 1.30	*		91.80 ± 1.19	0.22^o	87.80 ± 1.88	0.00^-	92.29 ± 0.95	0.03^+
Ring	88.23 ± 0.99	*		84.76 ± 1.81	0.00^-	88.16 ± 1.25	0.89^o	88.35 ± 1.42	0.83^o
Gamma	82.08 ± 0.85	*		82.49 ± 0.60	0.23^o	82.29 ± 1.09	0.64^o	81.99 ± 0.76	0.81^o
Satimage	85.52 ± 1.82	*		85.41 ± 1.06	0.87^o	85.02 ± 1.19	0.48^o	85.17 ± 1.46	0.65^o
Musk2	95.23 ± 0.97	*		94.88 ± 1.00	0.44^o	96.51 ± 0.85	0.01^+	96.83 ± 0.71	0.00^+
Shuttle	99.97 ± 0.02	*		99.96 ± 0.02	0.57^o	99.93 ± 0.05	0.04^-	99.97 ± 0.02	0.87^o
Sensorless	95.95 ± 2.68	*		98.79 ± 0.12	0.00^+	#		98.41 ± 0.17	0.01^+
MiniBooNE	88.04 ± 1.02	*		89.59 ± 0.25	0.00^+	#		89.57 ± 0.20	0.00^+
Winnipeg	98.45 ± 0.10	*		*		#		98.91 ± 0.04	0.00^+
Average (NOD)	90.38 (17)	85.30 (7)		89.35 (16)		89.63 (14)		90.85 (17)	
CAverage$^{\$}$	90.81	85.30		89.20		90.53		90.93	
Lose/Win/Tie		4/1/2		3/2/11		3/2/9		0/6/11	

NOD: Number of datasets over which the average is computed (indicated in bracket).

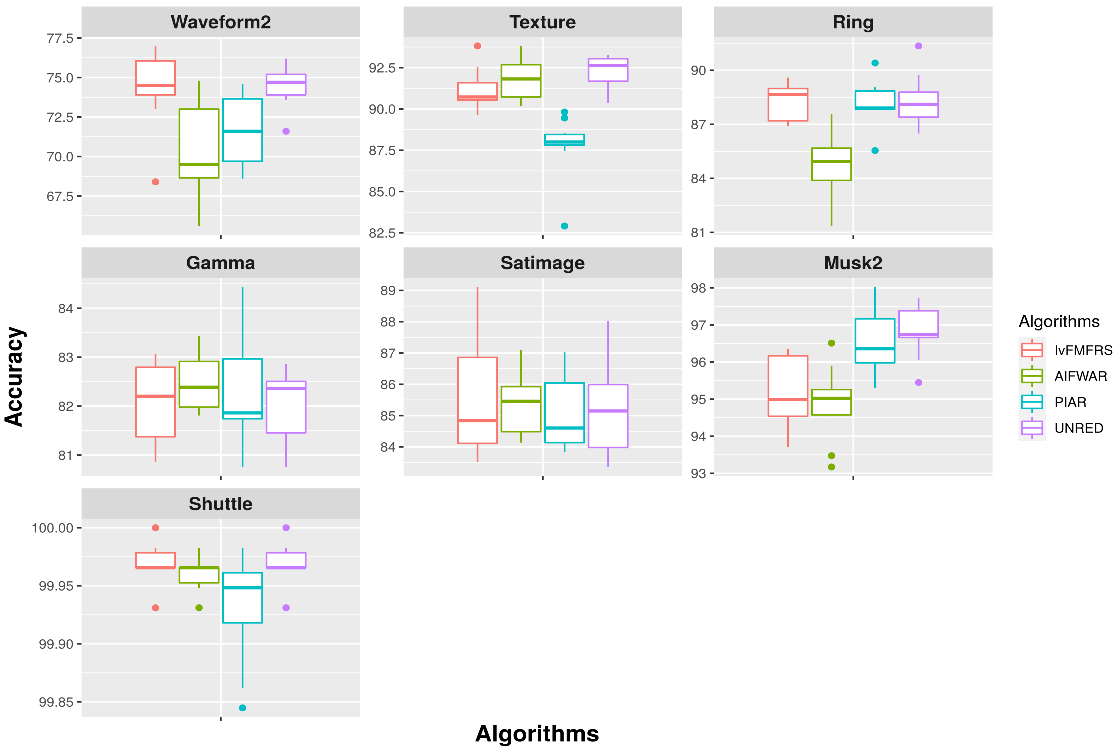
$: Average mean value over 7 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



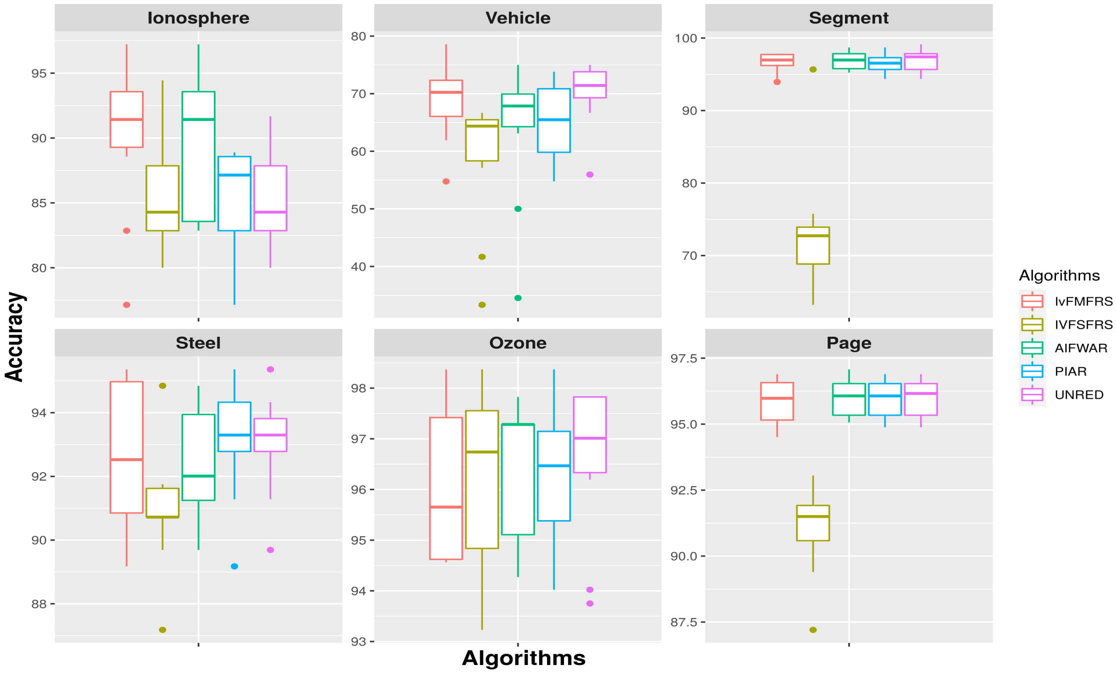
(b) Datasets having classification results by IvFMFRS, AIFWAR, PIAR and UNRED

Figure 6.2: Boxplot for Classification Accuracies Results with CART of Table 6.5

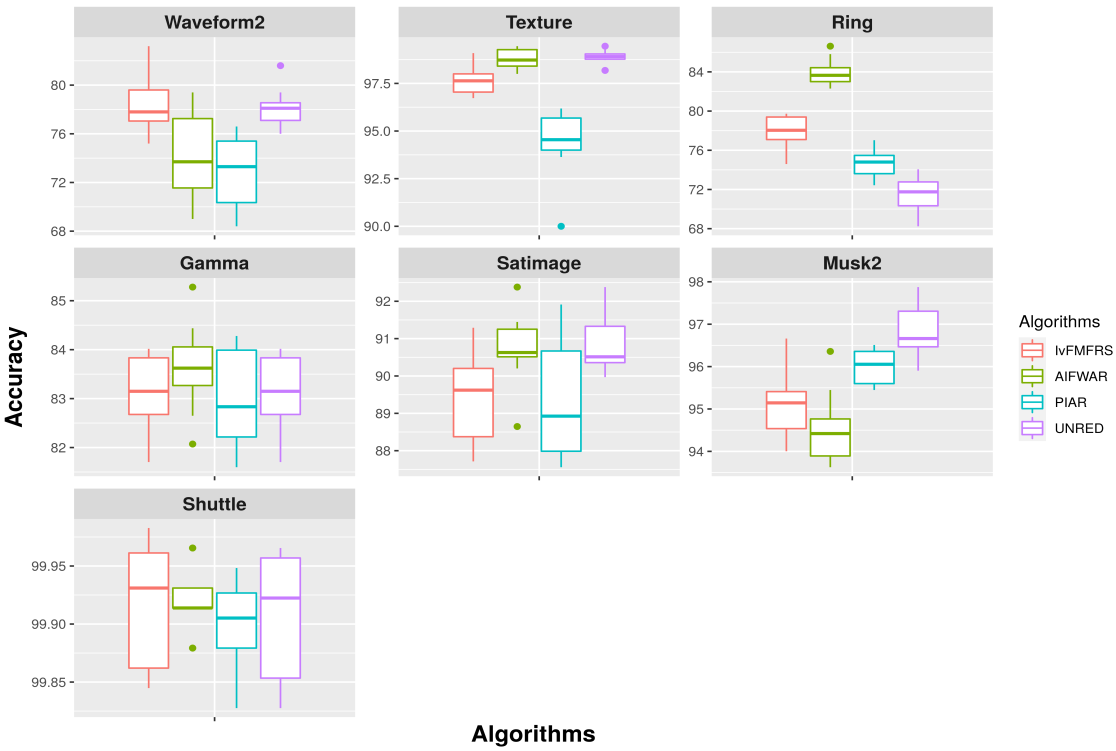
Table 6.6: Classification Accuracies Results (%) with kNN (k=3) in 10-FCV

Datasets	IvFMFRS	IV-FS-FRS		AIFWAR		PIAR		UNRED	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	90.29 ± 6.21	85.73 ± 5.25	0.09^{o}	89.44 ± 5.41	0.75^{o}	85.17 ± 4.25	0.05^{-}	85.17 ± 3.56	0.04^{-}
Vechile	68.68 ± 6.69	58.59 ± 11.60	0.03^{-}	63.92 ± 12.43	0.30^{o}	64.89 ± 6.90	0.23^{o}	70.21 ± 5.72	0.59^{o}
Segment	96.58 ± 1.42	73.46 ± 8.66	0.00^{-}	96.88 ± 1.19	0.61^{o}	96.49 ± 1.31	0.89^{o}	96.88 ± 1.55	0.65^{o}
Steel	92.63 ± 2.29	90.93 ± 1.90	0.09^{o}	92.32 ± 1.72	0.74^{o}	93.15 ± 1.83	0.59^{o}	93.05 ± 1.59	0.65^{o}
Ozone	96.27 ± 1.21	96.49 ± 1.29	0.70^{o}	96.59 ± 1.63	0.62^{o}	96.27 ± 1.44	0.68^{o}	96.33 ± 1.33	0.92^{o}
Page	95.81 ± 0.86	91.06 ± 1.73	0.00^{-}	96.03 ± 0.73	0.55^{o}	96.00 ± 0.73	0.62^{o}	96.02 ± 0.75	0.58^{o}
Robot	89.15 ± 1.53	87.57 ± 0.87	0.01^{-}	92.14 ± 1.13	0.00^{+}	87.57 ± 0.87	0.01^{-}	87.57 ± 0.87	0.01^{-}
Waveform2	78.40 ± 2.29	*		74.00 ± 3.71	0.01^{-}	72.72 ± 3.05	0.00^{-}	78.12 ± 1.59	0.75^{o}
Texture	97.71 ± 0.78	*		98.80 ± 0.54	0.00^{+}	94.42 ± 1.81	0.00^{-}	98.85 ± 0.41	0.00^{+}
Ring	77.80 ± 1.86	*		83.96 ± 1.36	0.00^{+}	74.61 ± 1.38	0.00^{-}	71.51 ± 1.77	0.00^{-}
Gamma	83.09 ± 0.83	*		83.64 ± 0.90	0.18^{o}	83.00 ± 0.99	0.83^{o}	83.09 ± 0.83	1.00^{o}
Satimage	89.46 ± 1.20	*		90.71 ± 0.97	0.02^{+}	89.32 ± 1.57	0.83^{o}	90.83 ± 0.73	0.01^{+}
Musk2	95.15 ± 0.81	*		94.54 ± 0.84	0.12^{o}	96.00 ± 0.43	0.01^{+}	96.82 ± 0.64	0.00^{+}
Shuttle	99.92 ± 0.06	*		99.92 ± 0.03	0.80^{o}	99.90 ± 0.04	0.45^{o}	99.91 ± 0.05	0.73^{o}
Sensorless	94.71 ± 4.21	*		99.35 ± 0.09	0.00^{+}	#		99.03 ± 0.14	0.00^{+}
MiniBooNE	90.01 ± 0.84	*		92.17 ± 0.13	0.00^{+}	#		92.20 ± 0.13	0.00^{+}
Winnipeg	99.49 ± 0.35	*		*		#		99.60 ± 0.04	0.35^{o}
Average (NOD)	90.04 (17)	83.40 (7)		90.42 (16)		88.24 (14)		89.83 (17)	
***CAverage*$^{\$}$**	89.91	83.40		89.61		88.50		89.31	
Lose/Win/Tie		4/0/3		1/6/9		5/1/8		3/5/9	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 7 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
\# represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



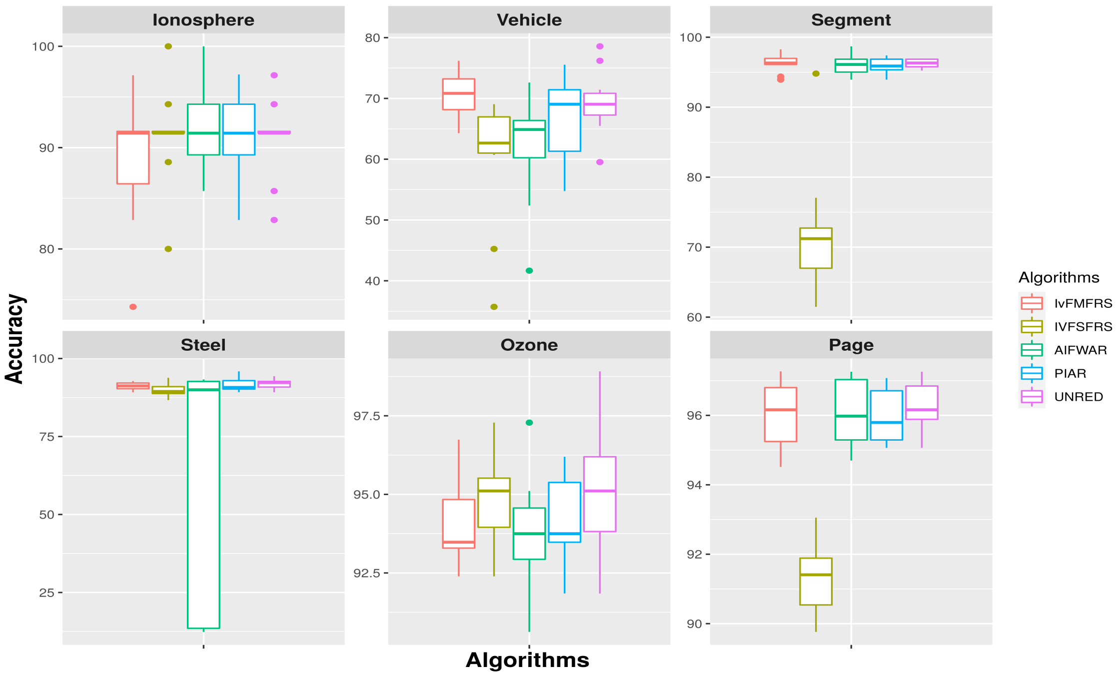
(b) Datasets having classification results by IvFMFRS, AIFWAR, PIAR and UNRED

Figure 6.3: Boxplot for Classification Accuracies Results with kNN of Table 6.6

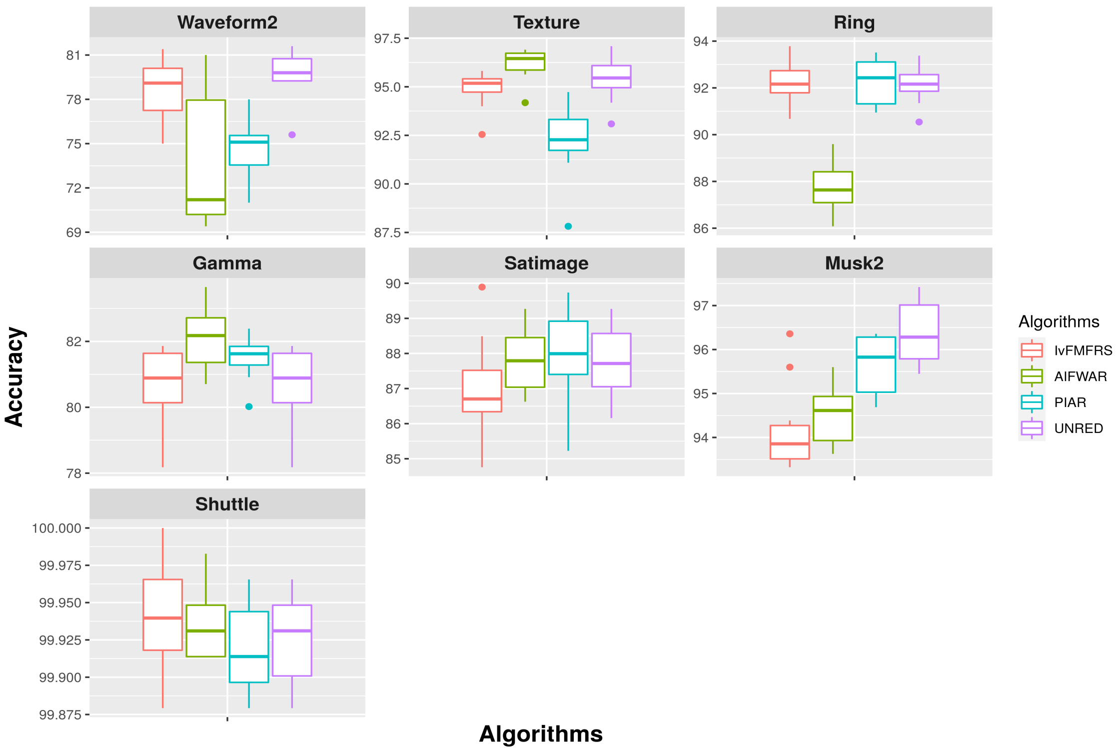
Table 6.7: Classification Accuracies Results (%) with kNN-FMNN in 10-FCV

Datasets	**IvFMFRS**	**IV-FS-FRS**		**AIFWAR**		**PIAR**		**UNRED**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	88.88 ± 6.53	91.17 ± 4.94	0.39^o	92.00 ± 4.63	0.23^o	91.44 ± 4.68	0.33^o	90.88 ± 4.00	0.42^o
Vechile	70.41 ± 4.18	59.79 ± 10.80	0.01^-	61.96 ± 8.94	0.01^-	66.37 ± 7.60	0.16^o	69.40 ± 5.33	0.64^o
Segment	96.23 ± 1.28	72.21 ± 9.05	0.00^-	96.06 ± 1.45	0.78^o	95.93 ± 1.08	0.57^o	96.23 ± 0.68	1.00^o
Steel	91.09 ± 1.25	89.75 ± 2.03	0.09^o	60.47 ± 40.65	0.03^-	91.55 ± 2.14	0.56^o	91.81 ± 1.49	0.26^o
Ozone	93.84 ± 1.52	95.73 ± 1.08	0.00^+	94.37 ± 1.72	0.47^o	94.16 ± 1.47	0.80^o	93.84 ± 1.76	1.00^o
Page	96.00 ± 0.97	91.30 ± 1.15	0.00^-	96.07 ± 1.00	0.87^o	95.98 ± 0.80	0.96^o	96.25 ± 0.70	0.51^o
Robot	93.75 ± 0.57	93.16 ± 1.03	0.13^o	91.55 ± 1.26	0.00^-	93.16 ± 1.03	0.13^o	93.16 ± 1.03	0.13^o
Waveform2	78.80 ± 2.07	*		73.56 ± 4.62	0.00^-	74.84 ± 2.07	0.00^-	9.70 ± 1.69	0.30^o
Texture	94.85 ± 0.97	*		96.18 ± 0.84	0.00^+	92.16 ± 1.86	0.00^-	95.36 ± 1.14	0.30^o
Ring	92.16 ± 0.94	*		87.80 ± 1.05	0.00^-	92.30 ± 0.98	0.76^o	92.14 ± 0.81	0.95^o
Gamma	80.74 ± 1.13	*		82.13 ± 0.99	0.01^+	81.48 ± 0.65	0.09^o	80.74 ± 1.13	1.00^o
Satimage	87.06 ± 1.40	*		87.80 ± 0.93	0.18^o	87.86 ± 1.48	0.23^o	87.79 ± 0.98	0.19^o
Musk2	94.18 ± 1.01	*		94.51 ± 0.63	0.39^o	95.67 ± 0.68	0.00^+	96.38 ± 0.76	0.00^+
Shuttle	99.94 ± 0.04	*		99.94 ± 0.02	0.81^o	99.92 ± 0.03	0.15^o	99.92 ± 0.03	0.32^o
Sensorless	94.73 ± 4.04	*		99.25 ± 0.10	0.00^+	#		98.48 ± 0.11	0.01^+
MiniBooNE	88.31 ± 0.70	*		89.40 ± 0.21	0.00^+	#		89.33 ± 0.27	0.00^+
Winnipeg	98.69 ± 0.49	*		*		#		98.07 ± 0.11	0.00^-
Average (NOD)	90.51 (17)	84.73 (7)		86.89 (16)		89.83 (14)		91.13 (17)	
***CAverage*$^{\$}$**	90.02	84.73		84.64		89.67		90.22	
Lose/Win/Tie		3/1/3		5/4/7		2/1/11		1/3/13	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 7 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.



(a) Datasets having classification results by all algorithms



(b) Datasets having classification results by IvFMFRS, AIFWAR, PIAR and UNRED

Figure 6.4: Boxplot for Classification Accuracies Results with kNN-FMNN of Table 6.7

Table 6.8: Computational Time Results (in Seconds) in 10-FCV

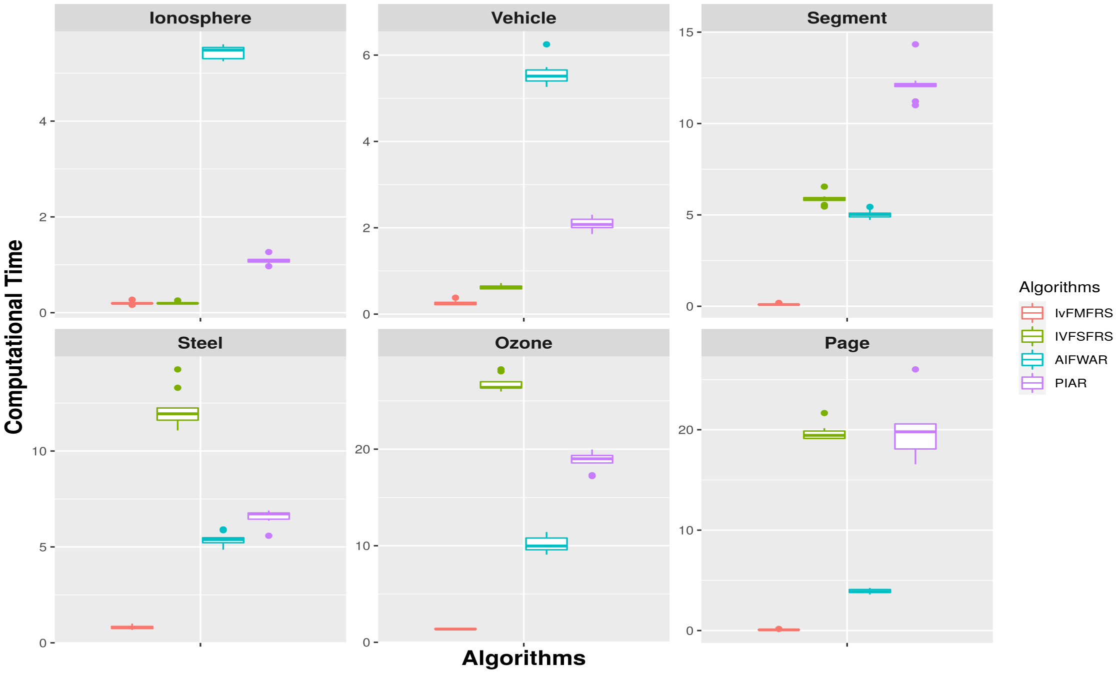
Datasets	**IvFMFRS**	**IV-FS-FRS**		**AIFWAR**		**PIAR**	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	0.20 ± 0.03	0.20 ± 0.02	0.73^{o}	5.44 ± 0.13	0.00^{-}	1.07 ± 0.09	0.00^{-}
Vechile	0.26 ± 0.05	0.63 ± 0.05	0.00^{-}	5.57 ± 0.28	0.00^{-}	2.09 ± 0.14	0.00^{-}
Segment	0.10 ± 0.03	5.87 ± 0.29	0.00^{-}	5.04 ± 0.21	0.00^{-}	12.15 ± 0.88	0.00^{-}
Steel	0.81 ± 0.10	12.17 ± 0.95	0.00^{-}	5.40 ± 0.32	0.00^{-}	6.56 ± 0.38	0.00^{-}
Ozone	1.37 ± 0.13	26.73 ± 2.19	0.00^{-}	11.00 ± 0.99	0.00^{-}	18.79 ± 0.90	0.00^{-}
Page	0.08 ± 0.03	19.67 ± 0.79	0.00^{-}	3.92 ± 0.20	0.00^{-}	19.80 ± 2.61	0.00^{-}
Robot	22.64 ± 3.06	50.11 ± 4.31	0.00^{-}	18.81 ± 1.84	0.00^{-}	70.35 ± 1.27	0.00^{-}
Waveform2	484.66 ± 32.08	*		32.23 ± 1.85	0.00^{+}	421.01 ± 40.23	0.00^{-}
Texture	0.33 ± 0.04	*		13.15 ± 0.95	0.00^{-}	84.68 ± 7.47	0.00^{-}
Ring	5.67 ± 0.42	*		12.47 ± 1.18	0.00^{-}	334.65 ± 48.92	0.00^{-}
Gamma	1.63 ± 0.08	*		11.05 ± 0.15	0.00^{-}	436.36 ± 24.41	0.00^{-}
Satimage	3.02 ± 0.40	*		16.68 ± 0.24	0.00^{-}	267.69 ± 36.64	0.00^{-}
Musk2	163.33 ± 14.79	*		66.77 ± 3.38	0.00^{+}	3562.27 ± 96.61	0.00^{-}
Shuttle	0.25 ± 0.04	*		20.87 ± 1.62	0.00^{-}	2221.72 ± 69.81	0.00^{-}
Sensorless	0.41 ± 0.06	*		601.63 ± 30.77	0.00^{-}	#	
MiniBooNE	3052.49 ± 119.13	*		1950.72 ± 21.63	0.00^{+}	#	
Winnipeg	38080.01 ± 2949.87	*		*		#	
Average (NOD)	2459.83 (17)	16.48 (7)		173.79 (16)		532.79 (14)	
CAverage$^{\$}$	3.63	16.48		4.59		18.68	
Lose/Win/Tie		6/0/1		13/3/0		14/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).

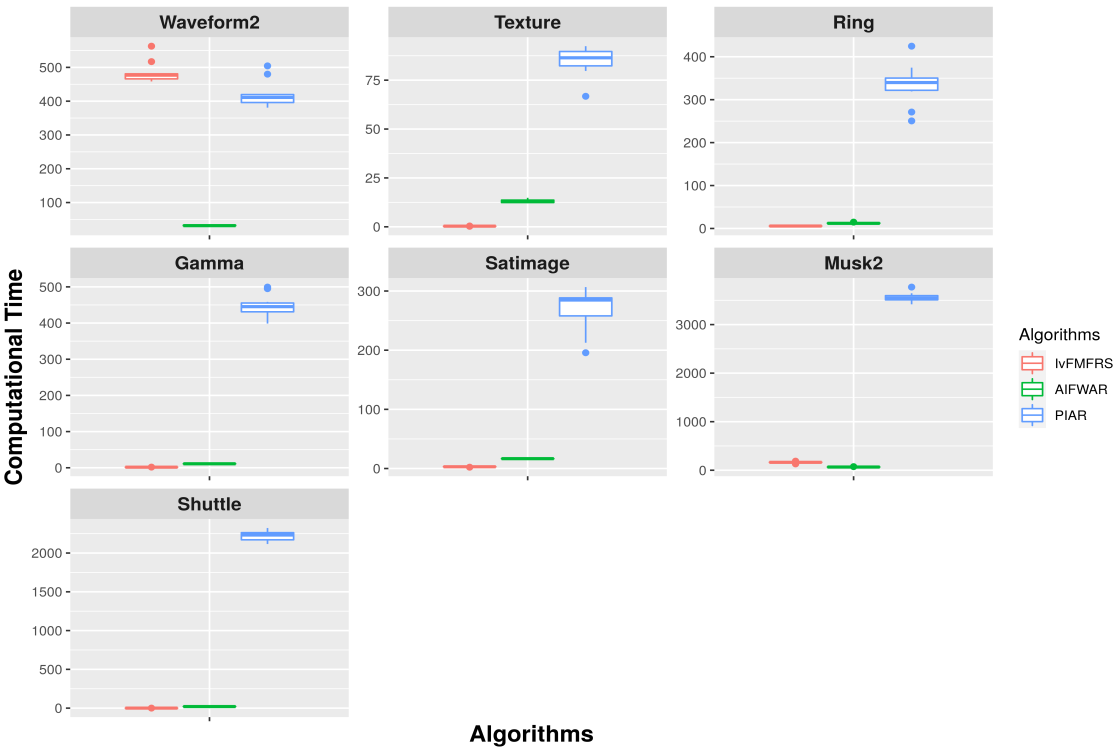
$: Average mean value over 7 datasets where all algorithms executed.

* represents non-executable due to memory overflow.

represents non-termination of program even after several hours.



(a) Datasets having computational time results by all algorithms



(b) Datasets having computational time results by IvFMFRS, AIFWAR and PIAR

Figure 6.5: Boxplot for Computational Time Results of Table 6.8

Table 6.9: Reduct Length Results in 10-FCV

Datasets	IvFMFRS	IV-FS-FRS		AIFWAR		PIAR	
	Mean ± Std	Mean ± Std	p-Val	Mean ± Std	p-Val	Mean ± Std	p-Val
Ionosphere	7.70 ± 0.48	8.60 ± 1.26	0.05^{-}	7.80 ± 2.10	0.88^{o}	17.30 ± 0.67	0.00^{-}
Vechile	12.90 ± 0.99	4.30 ± 1.25	0.00^{-}	9.40 ± 4.33	0.02^{+}	8.80 ± 0.79	0.00^{-}
Segment	7.90 ± 0.88	4.10 ± 3.51	0.00^{+}	10.40 ± 0.70	0.00^{-}	9.10 ± 0.32	0.00^{-}
Steel	12.00 ± 1.41	5.70 ± 2.00	0.00^{+}	6.30 ± 4.64	0.00^{+}	14.90 ± 0.32	0.00^{-}
Ozone	10.50 ± 0.85	19.80 ± 3.26	0.00^{-}	12.60 ± 7.37	0.38^{o}	28.50 ± 0.97	0.00^{-}
Page	7.50 ± 0.85	1.50 ± 0.53	0.00^{+}	7.50 ± 0.97	1.00^{o}	8.90 ± 0.32	0.00^{-}
Robot	12.70 ± 0.67	24.00 ± 0.00	0.00^{-}	4.10 ± 0.32	0.00^{+}	24.00 ± 0.00	0.00^{-}
Waveform2	13.80 ± 0.42	*		15.20 ± 17.12	0.80^{o}	24.80 ± 0.92	0.00^{-}
Texture	12.01 ± 1.05	*		15.90 ± 2.33	0.00^{-}	6.80 ± 0.42	0.00^{-}
Ring	15.80 ± 0.63	*		9.30 ± 0.67	0.00^{+}	17.90 ± 0.57	0.00^{-}
Gamma	10.00 ± 0.00	*		7.00 ± 0.00	0.00^{+}	9.00 ± 0.00	0.00^{-}
Satimage	15.60 ± 1.17	*		33.70 ± 4.30	0.00^{-}	13.30 ± 0.82	0.00^{-}
Musk2	16.30 ± 0.48	*		29.80 ± 23.03	0.08^{o}	71.70 ± 2.11	0.00^{-}
Shuttle	5.00 ± 0.67	*		5.10 ± 1.73	0.87^{o}	6.00 ± 0.00	0.00^{-}
Sensorless	10.60 ± 1.07	*		3.00 ± 0.00	0.00^{+}	#	
MiniBooNE	23.30 ± 1.06	*		47.70 ± 1.89	0.00^{-}	#	
Winnipeg	22.50 ± 1.06	*		*		#	
Average (NOD)	12.71 (17)	9.71 (7)		14.05 (16)		18.60 (14)	
***CAverage*$**	10.17	9.71		8.30		15.92	
Lose/Win/Tie		4/3/0		4/6/6		14/0/0	

NOD: Number of datasets over which the average is computed (indicated in bracket).
$: Average mean value over 7 datasets where all algorithms executed.
* represents non-executable due to memory overflow.
# represents non-termination of program even after several hours.

Analysis of Results

Classification accuracy results

Table 6.5, Table 6.6 and Table 6.7 show the classification results of CART, kNN and kNN-FMNN classifiers. In all classifiers, the CAverage value of the proposed algorithm IvFMFRS is higher than compared algorithms and very near to UNRED.

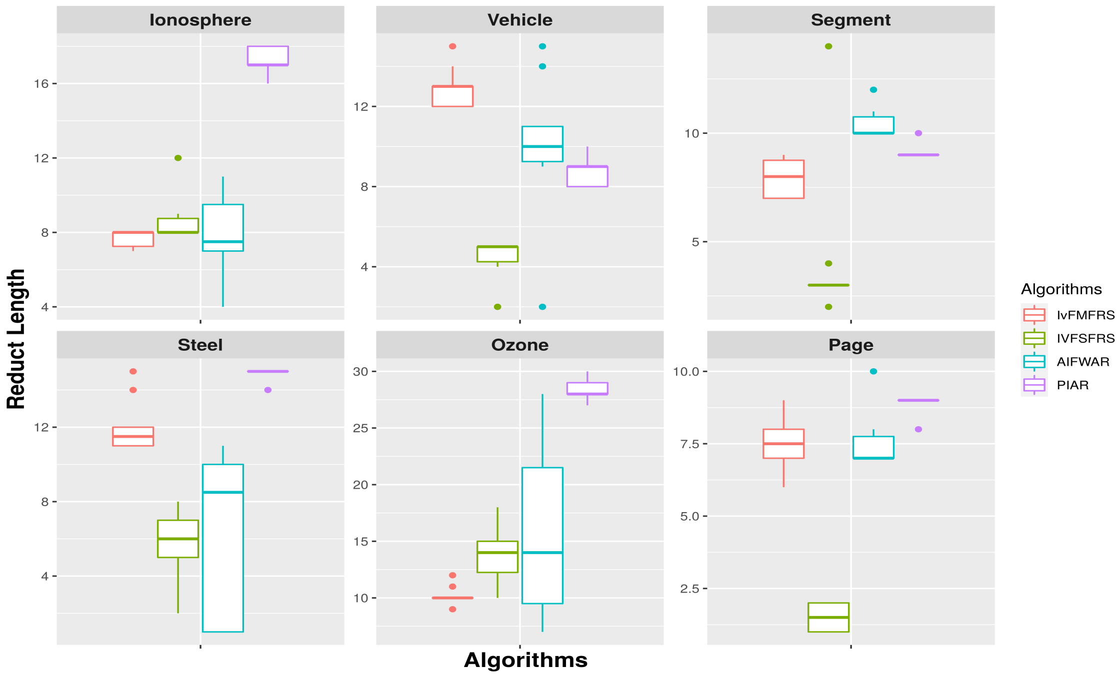
In Table 6.5, considering the overall 54 accuracy results across all the compared algorithms and UNRED in CART classifier, the cumulative lose/win/tie results are 10/11/33. In 33 classification results, the proposed algorithm IvFMFRS returned significantly similar results to compared algorithms and UNRED. Also, it is observed that wherever IvFMFRS performed a little inferior to compared algorithms and UNRED (i.e., 11 results), the differences in average mean are very small. In the remaining 10 results, the proposed algorithm IvFMFRS performed significantly better than the compared algorithms, and here also, it is observed that the difference in mean value is small.

Similarly, in other kNN and kNN-FMNN classifiers, as given in Table 6.6 and Table 6.7, majorly all algorithms performed statistically similar to each other. The cumulative lose/win/tie results in kNN classifier is 13/12/29 and in kNN-FMNN is 11/9/34. The further observation analysis details are given below.

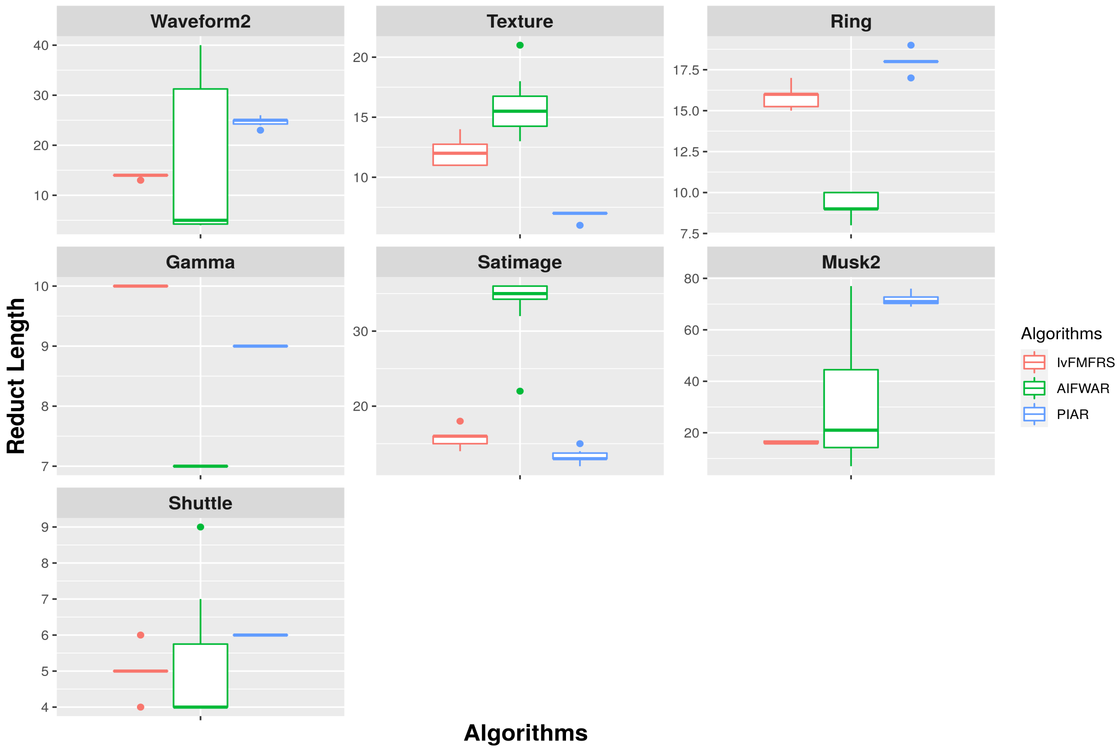
IvFMFRS achieved statistically better than AIFWAR and PIAR algorithms in Waveform2 dataset in all classifiers.

Based on CART classifier results in Table 6.5 and Fig. 6.2, in Robot and Ring datasets, IvFMFRS performed statistically significant than AIFWAR. Moreover, IvFMFRS obtained better in classification than IV-FS-FRS in Vehicle, Segment, Steel and Page datasets. However, in Robot and Musk2, IvFMFRS performed statistically inferior to PIAR, although the difference in average classification accuracies for both algorithms is insignificant. A similar case for AIFWAR algorithm, where it performed better than IvFMFRS in Robot and Ring datasets.

Similar conclusions can be obtained in the kNN and kNN-FMNN classifiers from Table 6.5 and Table 6.7 and their respective Fig. 6.3 and Fig. 6.4. In both classifiers, IvFMFRS achieved statistically significant than IV-FS-FRS in Vehicle, Segment, and Steel datasets. In Robot, Texture and Ring datasets, PIAR performed statistically inferior to IvFMFRS in kNN classifiers. Also, PIAR could not be able to compute reduct in a reasonable amount of time in Sensorless, MiniBooNE and Winnipeg datasets, where IvFMFRS algorithm could compute reduct with comparative classification accuracies. Moreover, IvFMFRS showed statistically equivalent to other algorithms and UNRED in most datasets. AIFWAR performed significantly better in Texture dataset than IvFMFRS, but the difference in mean value is very



(a) Datasets having reduct length results by all algorithms



(b) Datasets having reduct length results by IvFMFRS, AIFWAR and PIAR

Figure 6.6: Boxplot for Reduct Length Results of Table 6.9

less.

Eventually, it can be seen that the idea of computing the approximate reduct by IvFMFRS is satisfactory and effective in terms of classification results in given classifiers. As we can see, the average value of the individual mean of classification accuracy of the IvFMFRS algorithm for all datasets is quite similar to AIFWAR, PIAR, FDM-FMFRS and UNRED.

It is further observed that, in Waveform2, Gamma, Texture, Ring, Satimage, Musk2 and Shuttle datasets, IV-FS-FRS could not obtain reduct due to memory overflow at given system configuration where IvFMFRS, AIFWAR obtained reduct in reasonable computational time. A similar case happened for PIAR, IvFMFRS got reduct in Sensorless and MiniBooNE where PIAR could not. This is due to the aspect of representative instances in AIFWAR and fuzzy hyperboxes based granularization in IvFMFRS, achieving a significant reduction in space utilization. In Winnipeg dataset, IvFMFRS could compute reduct, whereas all compared algorithms could not.

Computational time results

In terms of computational times, as shown in Table 6.8 and Fig. 6.5, IvFMFRS incurred significantly less computational time than compared incremental algorithms (IV-FS-FRS, AIFWAR and PIAR) for all datasets except for Waveform2, Musk2 and MiniBooNE datasets. The proposed method IvFMFRS obtained the lowest CAverage value (3.63 seconds) on datasets, whereas compared algorithms and UNRED with CAverage showed a range between 5 and 19 seconds and evidently, seen that the cumulative lose/win/tie results of compared algorithms w.r.t. IvFMFRS are 33/3/1.

These substantial reductions in computational time of IvFMFRS are due to the dealing with hyperboxes constructed by the FMNN model where $|HBS| << |U|$. Thus, the speed-up computation and performance demonstrate the potential of the IvFMFRS algorithm and its suitability for larger datasets.

However, in Waveform2 and Musk2 datasets, IvFMFRS obtained statistically higher computational time than compared algorithms. Because, in each subset arrival, a large number of existing hyperboxes get updated or new hyperboxes created results in their corresponding entries in fuzzy DM are also updated and newly entered on each arrival. This way increased the substantial amount of computation time of IvFMFRS. Generally, in IvFMFRS, on each subset arrival, only a few hyperboxes are updated which results in less updation of entries in fuzzy DM. Because of this reason, IvFMFRS incurred less computation in most of the datasets.

The average mean value of IvFMFRS on overall datasets is 2459.83 seconds which is higher than compared algorithms. Because considering Winnipeg dataset results in average individual mean results higher than others, where our proposed algorithm could run on Winnipeg dataset where compared algorithms could not. None of the compared algorithms could scale to Winnipeg datasets. In all datasets, the resulting standard deviation of computation time presented very less variation, thus showing that the methodology is reliable as compared to others.

Reduct length results

From the results on reduct length shown in Table 6.9 and Fig. 6.6, IvFMFRS performed statistically significant, which means computed relevant attributes with smaller reduct size than IV-FS-FRS, AIFWAR and PIAR algorithms in most of the datasets and evidently seen that the cumulative lose/win/tie results of compared algorithms are 22/9/6. IvFMFRS performed statistically inferior in terms of reduct size from IV-FS-FRS in some datasets. But, the quality of reduct from IV-FS-FRS algorithm in terms of average classification accuracy is statistically inferior to IvFMFRS. Even IvFMFRS achieved statistically better than PIAR in all datasets. The average individual mean of IvFMFRS is lower than AIFWAR and PIAR and higher than IV-FS-FRS.

In summary, the relevance of IvFMFRS is significantly validated as it computes incremental reduct with lesser length and incurs less computational time while preserving similar or better classification accuracies than compared incremental approaches in most of the time.

6.5.4 Comparative Analysis of Incremental Reduct Algorithms

This section investigates the comparative analysis of the incremental algorithms (IvFMFRS, IV-FS-FRS, AIFWAR and PIAR) in aspects of reduct length and computational time in the incremental step of reduct computation. We are presenting two figures (Fig. 6.7 and Fig. 6.8) depicting the detailed change of the computational time and reduct size of IvFMFRS, IV-FS-FRS, AIFWAR and PIAR with subset continuously entering. Each dataset is randomly partitioned into the ten equal subsets for an experiment. We incrementally update an algorithm with a subset in each iteration to learn and find the corresponding approximate reduct. Here, we are depicting the cumulative computational time till that iteration and reduct size at that iteration in Fig. 6.7 and Fig. 6.8 respectively.

In both figures, the x-axis represents the sequence size of the data. And, the y-axis represents the computational time (in seconds) in Fig. 6.7 and reduct length in Fig. 6.8. The dashed line shows the results of IvFMFRS; the dotted line shows IVFSFRS; the solid line shows AIFWAR in figures. This experiment is conducted on only thirteen datasets, out of which in seven datasets, IV-FS-FRS could not compute the reduct on the given system. Both figures illustrate the efficiency of incremental algorithms on arriving subset sequences one by one.

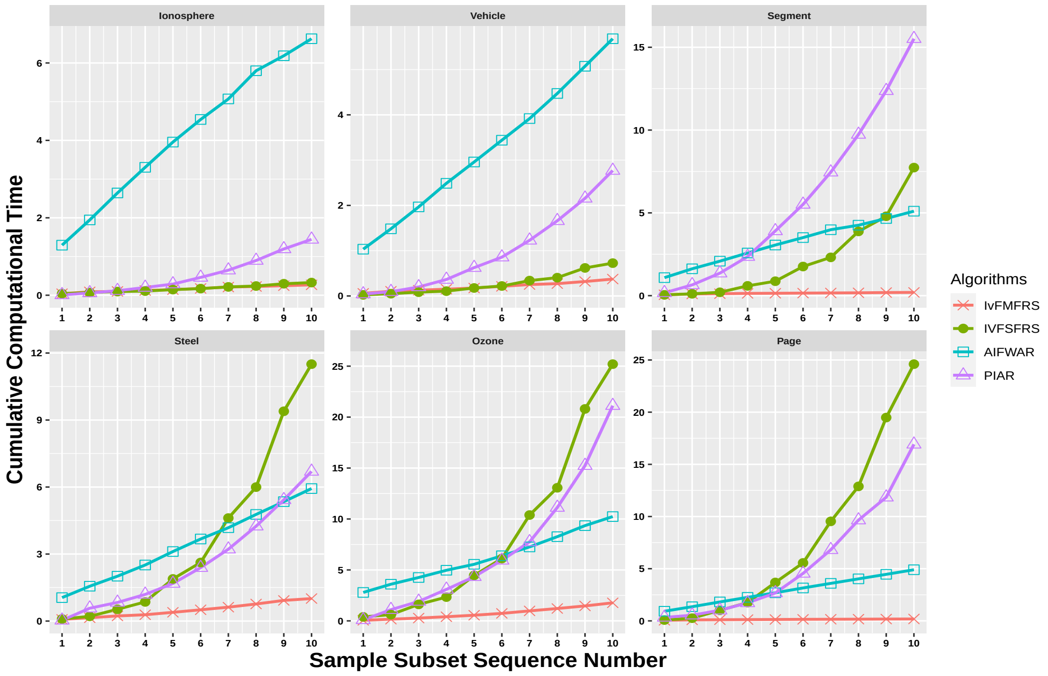
Analysis of Results

In Fig. 6.7, the computational time starts with base reduct computation from the first base part, and the rest of the timestamps are the time that is incurred for updating the reduct when a next sample subset has arrived. It can be seen from Fig. 6.7 that in most of the given datasets, the computational time for each subsequent sample as the number of samples increases result in a significant increase in computational time for both IV-FS-FRS and PIAR algorithms. However, in IvFMFRS and AIFWAR, the computational time is showing almost like a flat line for most datasets, indicating a roughly negligible amount of time is incurred when a subsequent sample is added after the base reduct computation on U_1.

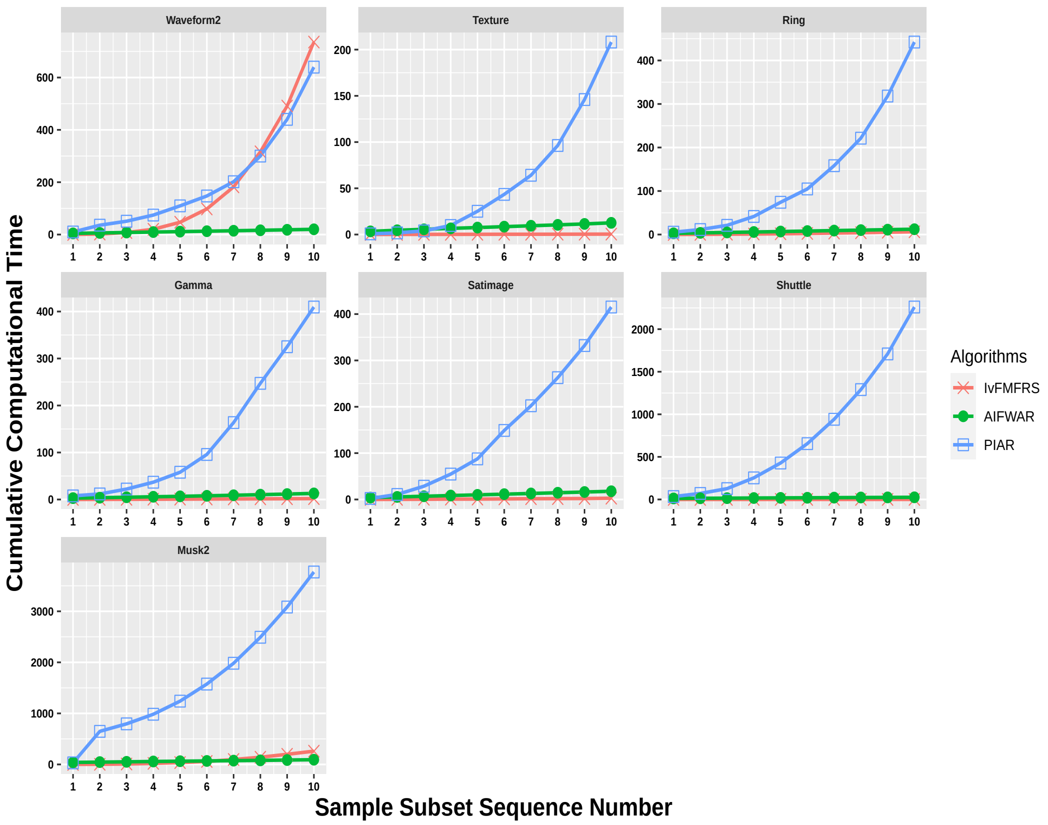
In IvFMFRS, the changes that have happened to the fuzzy DM and computational effort are actually very much minimal when it comes to our proposed algorithm. The size of fuzzy DM signifies the computational time that is involved when a new subset is added. This process is further attributed to utilizing FMNN as a preprocessor in fuzzy DM construction. FMNN is absorbing many new objects accommodated into the existing hyperboxes result in no changes in fuzzy DM, and the changes that are happening almost equivalent when the subsequent subset is added. Hence, the computation effort seems very small (near to zero) in each subsequent step after the first step. Usually, changes in FMNN tend to update in fuzzy DM construction. So in our case, the fuzzy DM size is not significantly growing from one sample to subsequent sample arriving in most datasets. However, in respective algorithms, fuzzy DM (especially in IV-FS-FRS) sizes are significantly changing, increasing computation time as the sample subset is growing.

In Waveform2, IvFMFRS shows a significant increase in computational time for each subsequent sample arrival. Because, in each arrival, almost all existing hyperboxes are updated, or new hyperboxes are created to accommodate objects, results in many changes and update in their entries in fuzzy DM. These substantial changes in hyperboxes and their corresponding entries in fuzzy DM in each subsequent sample arrival impacts significant growth in computational time.

From Fig. 6.8, it can be seen that both AIFWAR and IV-FS-FRS reduct exhibit a significant fluctuation in reduct size when a new sample is added. However, in IvFMFRS, the change in reduct size is very gradual, and it goes from a smaller reduct length to a little bigger length as the sample subset arrives. This gradual increase in reduct size is perhaps due to the

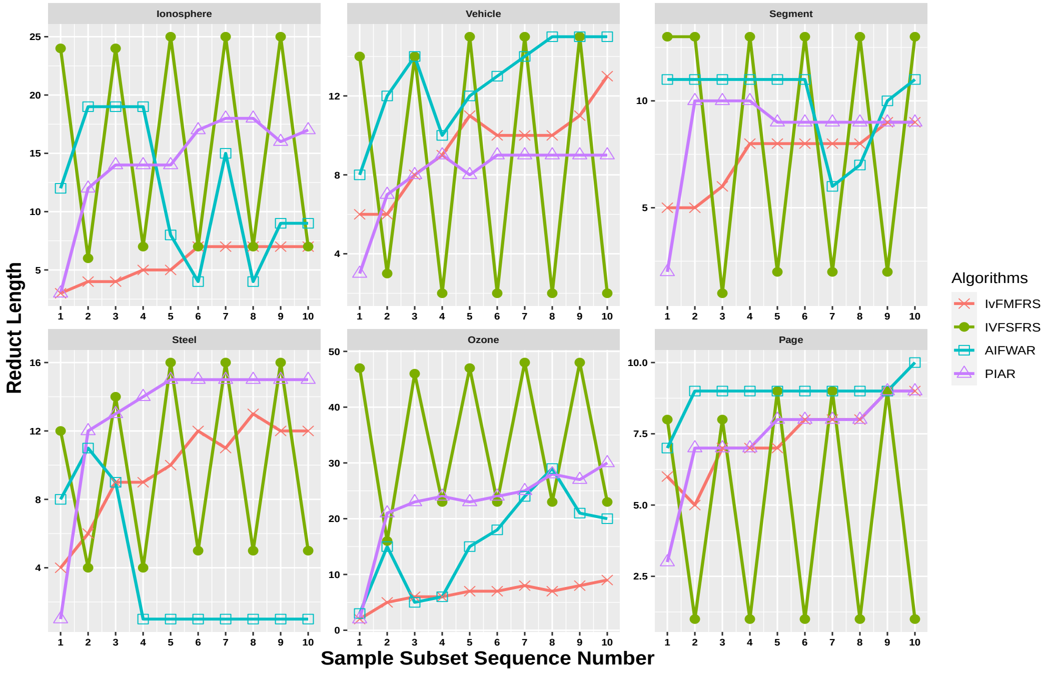


(a) Datasets having cumulative computational time results by all algorithms

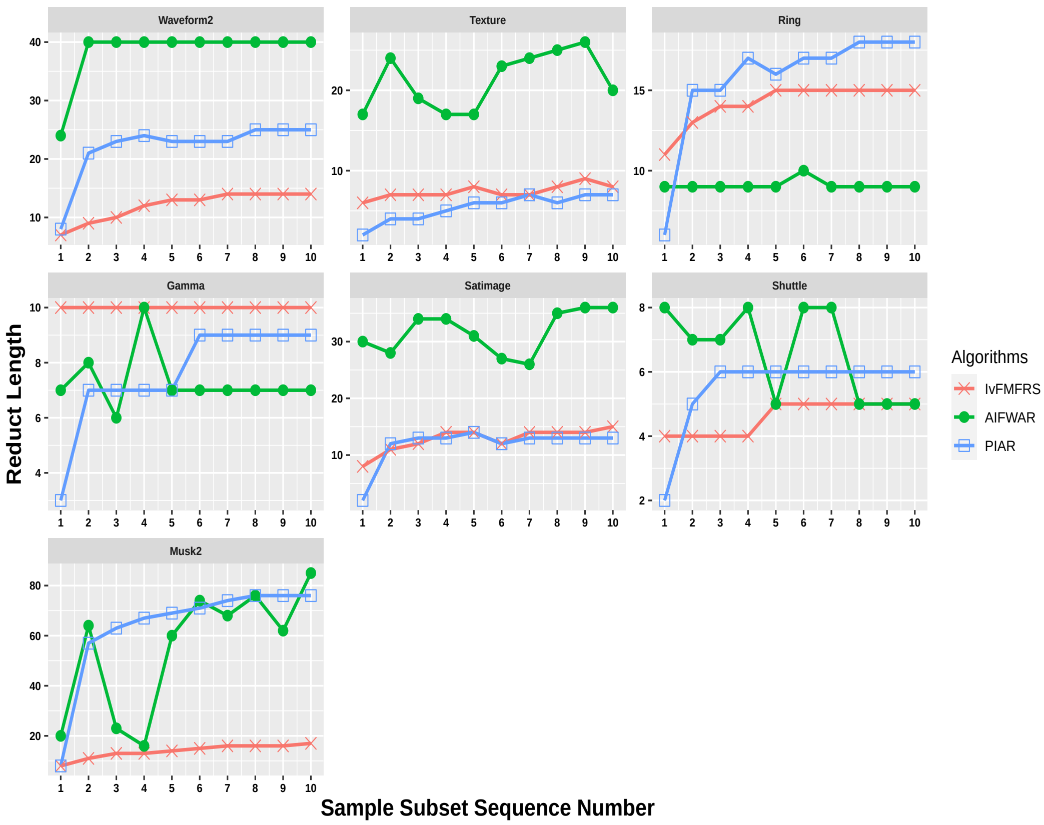


(b) Datasets having cumulative computational time results by IvFMFRS, AIFWAR and PIAR

Figure 6.7: Cumulative Computational Results



(a) Datasets having reduct length results by all algorithms



(b) Datasets having reduct length results by IvFMFRS, AIFWAR and PIAR

Figure 6.8: Reduct Length Results

following reason. In IvFMFRS algorithm, whenever a sample is entered, the SFS algorithm adds new attributes in the existing reduct, followed by the SBE algorithm to remove redundant attributes. In the AIFWAR algorithm, the attributes that are included in the existing reduct through the SFS process are followed by the wrapper technique for searching for the best attribute subset in reduct.

So, in the case of SBE inclusion or wrapper technique, removing most of the earlier present reduct attributes exhibits a lot of variance in reduct size, which can be seen in IV-FS-FRS and AIFWAR algorithms in Fig. 6.8. But in our case, the change is not much significant and not much variation in the reduct size observed. The attributes that are added in our approach are significant even after new attributes are included in the SFS process, which is getting retained in SBE process. As in our approach, attributes are selected based on discernibility over hyperboxes, which represent a set of objects of the decision system, leading to the selection of highly significant attributes. This aspect of selecting significant attributes as part of the SFS process due to FMNN preprocessing is aiding in making very less oscillation in reduct length, as seen in Fig. 6.8.

6.6 Summary

The proposed IvFMFRS is an incremental adaptation of FDM-FMFRS for incremental reduct computation using FRS. The incremental updation of reduct with the onset of new training data involves three phases: updation of hyperboxes through FMNN to include new training patterns, updation of fuzzy DM based on updated hyperboxes and update current reduct using SFS strategy followed by SBE strategy. FMNN preprocessing results in relatively fewer changes to the discernibility matrix than object-based, resulting in IvFMFRS being efficient from the aspects of both computational time and space utilization simultaneously. The detailed comparative experimental study is conducted with state of the art incremental FRS approaches and established the relevance of IvFMFRS in obtaining reduct with increased scalability and comparable or improved generalizability of the classifier models induced. It is also observed that the changes to the reduct in incremental learning of IvFMFRM are gradual in nature with better stability. IvFMFRS can scale to much larger datasets than the compared approaches. In the future, we will investigate distributed/parallel algorithms for IvFMFRS for achieving scalability to very large scale decision systems.

Chapter 7

Conclusions and Future Work

The primary objective of our research work is to explore the potential possibility of utilizing information granules in the form of hyperboxes and formulating algorithms for granular computing using hyperboxes in solving the standard problems of data mining and machine learning. Our research focuses on building hybrid soft computing models where fuzzy min-max neural network (FMNN) is one of the components, and the hyperboxes are utilized in other components to achieve the advantages of granular computing.

7.1 Conclusions

This section provides the brief conclusion of the contributions.

In Chapter 3, we proposed an algorithm kNN-FMNN as the hybridization of FMNN with kNN to overcome the contraction step in FMNN and enhance pattern classification. The comparative experiment was performed on kNN-FMNN with state-of-the-art FMNN approaches on several benchmark datasets. The experimental results established that kNN-FMNN achieved better classification accuracy than state-of-the-art FMNN algorithms in significantly less computational time in most datasets with a fewer number of hyperboxes. Also, we identified empirically that 0.3 is the appropriate value for parameter θ, which controls the size of the hyperbox.

In Chapter 4, we investigated fuzzy rough sets (FRS) approaches that provide a framework for reduct (feature subset selection) computation for decision systems. However, the existing FRS-based feature selection approaches are intractable for large decision systems due to the space complexity of the FRS methodology. We studied and proposed FDM-FMFRS as the hybridization of FMNN with FRS model for reduct computation, intending to increase scalability on benchmark datasets. The extensive experimental study was done on several benchmark datasets to establish the relevance of FDM-FMFRS reduct. Results demonstrated that

FDM-FMFRS achieved significant computational gains over existing state-of-the-art FRS approaches with similar or better classification accuracies and could scale to such large datasets where existing FRS algorithms are unable to compute due to space constraints.

In Chapter 5, we extended the FDM-FMFRS into a proposed algorithm (CDM-FMFRS) in terms of further scalability and improvised the reduct computation. Also, we enriched crisp discernibility relation with extended overlapping criteria and tolerance parameter. The comparative experiment was done on CDM-FMFRS with FDM-FMFRS and state-of-the-art FRS approaches. And results demonstrated that CDM-FMFRS achieved significant scalability against FDM-FMFRS but an increase in reduct size due to crisp formulation. Whenever possible, we recommend CDM-FMFRS as an alternative to FDM-FMFRS in a situation where FDM-FMFRS fails to obtain reduct owing to a memory overflow error.

In Chapter 6, we explored and proposed a scalable incremental reduct computation in FRS with FMNN preprocessing. IvFMFRS is an incremental adaptation of FDM-FMFRS. FMNN preprocessing resulted in relatively fewer changes to the discernibility matrix, resulting in IvFMFRS being efficient from aspects of computational time and space utilization simultaneously. The detailed comparative experimental study was conducted with state-of-the-art incremental FRS algorithms and established the relevance of IvFMFRS in obtaining reduct with increased scalability and comparable or improved generalizability of the classifier models induced. Also, the changes to the reduct in incremental learning in IvFMFRS were gradual in nature with better stability against compared algorithms.

7.2 Future Work

This section provides some insights into future work.

In the current scenario, Big data has gained much attention from every industry and made promising for business applications [24, 111]. Three aspects characterize big data, i.e., volume, variety and velocity [53]. The aspect of the velocity is dealt with in this thesis with our proposed incremental approaches to FRS reduct computation. We have made a significant achievement in the volume aspect through our proposed FDM-FMFRS and CDM-FMFRS approaches. To deal with the scenario when the hyperboxes-based representation of the discernibility matrix doesn't fit into single system memory, we will be proposing Apache Spark MapReduce-based adaptations of FDM-FMFRS and CDM-FMFRS in the future. Thus, our proposed work will deal with the volume characteristics of big data.

Due to the nature of FMNN, currently, our proposed approaches work only on numeric decision systems. As data comes from multiple sources and in multiple types, dealing with a variety of data will be a problem for our approaches. In the future, we plan to generalize our models on hybrid datasets that include both categorical and numeric attributes and apply the distributive framework to deal with big data scenarios. Thus, our proposed work will deal the variety characteristics of big data.

Currently, FMNN has another limitation in dealing with very large dimensional datasets. In such datasets, owing to the curse of dimensionality [7, 35, 60], the feasibility of meeting expansion criteria will become difficult, and hence FMNN may result in inducing a collection of point hyperboxes or hyperboxes containing very few objects. Such a scenario of finer granulation negates the advantages obtained by proposed approaches through FMNN. The extensive experimental results presented in the thesis have significantly demonstrated that both kNN-FMNN algorithm and FMNN preprocessing based FRS reduct computation approaches achieved significant computation gain without compromising on efficient classifier construction for datasets of large object space and moderate attribute space. We aim to work in the future to propose modifications of FMNN such that FMNN becomes suitable for very large dimensional datasets. This will help in extending the advantages of the proposed approaches to very large dimension datasets.

References

[1] Abdulghani Ali Ahmed and Mohammed Falah Mohammed. **SAIRF: A similarity approach for attack intention recognition using fuzzy min-max neural network**. *Journal of Computational Science*, **25**:467 – 473, 2018. ()

[2] Chandrashekhar Azad and Vijay Kumar Jha. **Fuzzy min–max neural network and particle swarm optimization based intrusion detection system**. *Microsystem Technologies*, **23**(4):907–918, Apr 2017. ()

[3] M. Banerjee, S. Mitra, and S. K. Pal. **Rough fuzzy MLP: knowledge encoding and classification**. *IEEE Transactions on Neural Networks*, **9**(6):1203–1216, 1998. ()

[4] Andrzej Bargiela, Witold Pedrycz, and Masahiro Tanaka. **An inclusion/exclusion fuzzy hyperbox classifier**. *KES Journal*, **8**:91–98, 08 2004. ()

[5] Ismat Beg and Samina Ashraf. **Similarity measures for fuzzy sets**. *Applied and Computational Mathematics*, **8**:192–202, 03 2009. ()

[6] Mininath R. Bendre and Vijaya R. Thool. **Analytics, challenges and applications in big data environment: a survey**. *Journal of Management Analytics*, **3**(3):206–239, 2016. ()

[7] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. **When Is "Nearest Neighbor" Meaningful?** *ICDT 1999. LNCS*, **1540**, 12 1997. ()

[8] Gail A. Carpenter and Stephen Grossberg. **A massively parallel architecture for a self-organizing neural pattern recognition machine**. *Computer Vision, Graphics, and Image Processing*, **37**(1):54–115, 1987. ()

[9] Chao-Ton Su and Jyh-Hwa Hsu. **An extended Chi2 algorithm for discretization of real value attributes**. *IEEE Transactions on Knowledge and Data Engineering*, **17**(3):437–441, 2005. ()

[10] D. Chen, L. Zhang, S. Zhao, Q. Hu, and P. Zhu. **A Novel Algorithm for Finding Reducts With Fuzzy Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **20**(2):385–389, 2012. ()

[11] Ming-Syan Chen, Jiawei Han, and P.S. Yu. **Data mining: an overview from a database perspective**. *IEEE Transactions on Knowledge and Data Engineering*, **8**(6):866–883, 1996. ()

[12] Chris Cornelis, Richard Jensen, Germán Hurtado, and Dominik Śle¸zak. **Attribute selection with fuzzy decision reducts**. *Information Sciences*, **180**(2):209–224, 2010. ()

[13] Chris Cornelis, Richard Jensen, Germán Hurtado, and Dominik Śle¸zak. **Attribute selection with fuzzy decision reducts**. *Information Sciences*, **180**(2):209–224, 2010. ()

[14] Corinna Cortes and Vladimir Vapnik. **Support-vector networks**. *Machine Learning*, **20**(3):273–297, Sep 1995. ()

[15] J. Dai, H. Hu, W. Wu, Y. Qian, and D. Huang. **Maximal-Discernibility-Pair-Based Approach to Attribute Reduction in Fuzzy Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **26**(4):2174–2187, 2018. ()

[16] Asit K. Das, Shampa Sengupta, and Siddhartha Bhattacharyya. **A group incremental feature selection for classification using rough set theory based genetic algorithm**. *Applied Soft Computing*, **65**:400 – 411, 2018. ()

[17] R. Davtalab, M. H. Dezfoulian, and M. Mansoorizadeh. **Multi-Level Fuzzy Min-Max Neural Network Classifier**. *IEEE Transactions on Neural Networks and Learning Systems*, **25**(3):470–482, 2014. ()

[18] Reza Davtalab, Mostafa Parchami, Mir Hossein Dezfoulian, Muharram Mansourizade, and Bahareh Akhtar. **M-FMCN: Modified Fuzzy Min-Max Classifier Using Compensatory Neurons**. In *Proceedings of the 11th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, AIKED'12, page 77–82, Stevens Point, Wisconsin, USA, 2012. World Scientific and Engineering Academy and Society (WSEAS). ()

[19] Supriya Kumar De and P.Radha Krishna. **Clustering web transactions using rough approximation**. *Fuzzy Sets and Systems*, **148**(1):131–138, 2004. Web Mining Using Soft Computing. ()

[20] S. Deshmukh and S. Shinde. **Diagnosis of Lung Cancer using Pruned Fuzzy Min-Max Neural Network**. In *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, pages 398–402, Sept 2016. ()

[21] Dheeru Dua and Casey Graff. **UCI Machine Learning Repository**, 2017. ()

[22] Didier Dubois and Henri Prade. **ROUGH FUZZY SETS AND FUZZY ROUGH SETS***. *International Journal of General Systems*, **17**(2-3):191–209, 1990. ()

[23] Didier Dubois and Henri Prade. *Putting Rough Sets and Fuzzy Sets Together*, pages 203–232. Springer Netherlands, Dordrecht, 1992. ()

[24] Danyel Fisher, Rob DeLine, Mary Czerwinski, and Steven Drucker. **Interactions with Big Data Analytics**. *Interactions*, **19**(3):50–59, may 2012. ()

[25] Evelyn Fix and J. L. Hodges. **Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties**. *International Statistical Review*, **57**(3):238–247, 1989. ()

[26] Yahya Forghani and Hadi Sadoghi Yazdi. **Fuzzy Min–Max Neural Network for Learning a Classifier with Symmetric Margin**. *Neural Processing Letters*, **42**(2):317–353, Oct 2015. ()

[27] Robert Fullér. *Introduction to neuro-fuzzy systems.* Advances in soft computing. Physica-Verlag, 2000. ()

[28] B. Gabrys and A. Bargiela. **General fuzzy min-max neural network for clustering and classification**. *IEEE Transactions on Neural Networks*, **11**(3):769–783, 2000. ()

[29] S. García, J. Luengo, J. A. Sáez, V. López, and F. Herrera. **A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning**. *IEEE Transactions on Knowledge and Data Engineering*, **25**(4):734–750, 2013. ()

[30] Alexander Gepperth and Barbara Hammer. **Incremental learning algorithms and applications**. In *European Symposium on Artificial Neural Networks (ESANN)*, pages 1–12, Bruges, Belgium, 2016. ()

[31] Soumen Ghosh, P. S. V. S. Sai Prasad, and C. Raghavendra Rao. **Third Order Backward Elimination Approach for Fuzzy-Rough Set Based Feature Selection**. In B. Uma Shankar, Kuntal Ghosh, Deba Prasad Mandal, Shubhra Sankar Ray, David Zhang, and Sankar K. Pal, editors, *Pattern Recognition and Machine Intelligence*, pages 254–262, Cham, 2017. Springer International Publishing. ()

[32] Salvatore Greco, Benedetto Matarazzo, and Roman Slowinski. **Fuzzy Similarity Relation as a Basis for Rough Approximations.** In Lech Polkowski and Andrzej Skowron, editors, *Rough Sets and Current Trends in Computing*, **1424** of *Lecture Notes in Computer Science*, pages 283–289. Springer, 1998. ()

[33] ROBERT HECHT-NIELSEN. **III.3 - Theory of the Backpropagation Neural Network**Based on "nonindent" by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.** In Harry Wechsler, editor, *Neural Networks for Perception*, pages 65–93. Academic Press, 1992. ()

[34] Qinghua Hu, Daren Yu, and Zongxia Xie. **Information-preserving hybrid data reduction based on fuzzy-rough techniques**. *Pattern Recognition Letters*, **27**(5):414–423, 2006. ()

[35] Piotr Indyk and Rajeev Motwani. **Approximate nearest neighbors: towards removing the curse of dimensionality**. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998. ()

[36] Paul Jaccard. **Nouvelles Recherches Sur la Distribution Florale**. *Bulletin de la Societe Vaudoise des Sciences Naturelles*, **44**:223–70, 01 1908. ()

[37] Balasubramaniam Jayaram and Radko Mesiar. **On special fuzzy implications**. *Fuzzy Sets and Systems*, **160**(14):2063–2085, 2009. Theme: Aggregation Operators. ()

[38] R. Jensen and Q. Shen. **Fuzzy-rough attribute reduction with application to web categorization**. *Fuzzy Sets and Systems*, **141**(3):469 – 485, 2004. ()

[39] R. Jensen and Q. Shen. **Semantics-preserving dimensionality reduction: rough and fuzzy-rough-based approaches**. *IEEE Transactions on Knowledge and Data Engineering*, **16**(12):1457–1471, 2004. ()

[40] R. Jensen and Q. Shen. **Fuzzy-Rough Sets Assisted Attribute Selection**. *IEEE Transactions on Fuzzy Systems*, **15**(1):73–89, Feb 2007. ()

[41] R. Jensen and Q. Shen. **New Approaches to Fuzzy-Rough Feature Selection**. *IEEE Transactions on Fuzzy Systems*, **17**(4):824–838, 2009. ()

[42] Richard Jensen and Neil Mac Parthalain. **Nearest Neighbour-Based Fuzzy-Rough Feature Selection.** In Chris Cornelis, Marzena Kryszkiewicz, Dominik Slezak, Ernestina Menasalvas Ruiz, Rafael Bello, and Lin Shang, editors, *RSCTC*, **8536** of *Lecture Notes in Computer Science*, pages 35–46. Springer, 2014. ()

[43] Richard Jensen and Neil Mac Parthaláin. **Towards scalable fuzzy-rough feature selection**. *Information Sciences*, **323**:1 – 15, 2015. ()

[44] Richard Jensen, A. Tuson, and Q. Shen. **Finding rough and fuzzy-rough set reducts with SAT**. *Inf. Sci.*, **255**:100–120, 2014. ()

[45] Richard Jensen, Andrew Tuson, and Qiang Shen. **Finding rough and fuzzy-rough set reducts with SAT**. *Information Sciences*, **255**:100–120, 2014. ()

[46] Yunge Jing, Tianrui Li, Junfu Huang, and Yingying Zhang. **An incremental attribute reduction approach based on knowledge granularity under the attribute generalization**. *International Journal of Approximate Reasoning*, **76**:80 – 95, 2016. ()

[47] Yunge Jing, Tianrui Li, Chuan Luo, Shi-Jinn Horng, Guoyin Wang, and Zeng Yu. **An incremental approach for attribute reduction based on knowledge granularity**. *Knowledge-Based Systems*, **104**:24 – 38, 2016. ()

[48] S. Kabir, C. Wagner, T. C. Havens, D. T. Anderson, and U. Aickelin. **Novel similarity measure for interval-valued data based on overlapping ratio**. In *Proceeding of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, 2017. ()

[49] Ho J. Kim, Tae W. Ryu, Thai T. Nguyen, Joon S. Lim, and Sudhir Gupta. **A Weighted Fuzzy Min-Max Neural Network for Pattern Classification and**

Feature Extraction. In Antonio Laganá, Marina L. Gavrilova, Vipin Kumar, Youngsong Mun, C. J. Kenneth Tan, and Osvaldo Gervasi, editors, *Computational Science and Its Applications – ICCSA 2004*, pages 791–798, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ()

[50] Ho-Joon Kim, Juho Lee, and Hyun-Seung Yang. **A Weighted FMM Neural Network and Its Application to Face Detection**. In Irwin King, Jun Wang, Lai-Wan Chan, and DeLiang Wang, editors, *Neural Information Processing*, pages 177–186, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ()

[51] Ho-Joon Kim and Hyun-Seung Yang. **A Weighted Fuzzy Min-Max Neural Network and Its Application to Feature Analysis**. In Lipo Wang, Ke Chen, and Yew Soon Ong, editors, *Advances in Natural Computation*, pages 1178–1181, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ()

[52] S. A. Kumar, A. Kumar, V. Bajaj, and G. K. Singh. **An Improved Fuzzy Min–Max Neural Network for Data Classification**. *IEEE Transactions on Fuzzy Systems*, **28**(9):1910–1924, 2020. ()

[53] Douglas Laney. **3D Data Management: Controlling Data Volume, Velocity, and Variety**. Technical report, META Group, February 2001. ()

[54] Daniel T. Larose and Chantal D. Larose. *An Introduction to Data Mining*, chapter 1, pages 1–15. John Wiley and Sons, Ltd, 2014. ()

[55] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. **Deep learning**. *Nature*, **521**(7553):436–444, May 2015. ()

[56] T. Y. Lin and N. Cercone. *Rough Sets and Data Mining: Analysis of Imprecise Data*. Kluwer Academic Publishers, USA, 1996. ()

[57] Pawan Lingras. **Comparison of neofuzzy and rough neural networks**. *Information Sciences*, **110**(3):207–215, 1998. ()

[58] Huan Liu and Rudy Setiono. **Feature Selection via Discretization**. *IEEE Trans. on Knowl. and Data Eng.*, **9**(4):642–645, July 1997. ()

[59] Ye Liu, Lidi Zheng, Yeliang Xiu, Hong Yin, Suyun Zhao, Xizhao Wang, Hong Chen, and Cuiping Li. **Discernibility matrix based incremental feature selection on fused decision tables**. *International Journal of Approximate Reasoning*, **118**:1 – 26, 2020. ()

[60] R. B. MARIMONT and M. B. SHAPIRO. **Nearest Neighbour Searches and the Curse of Dimensionality**. *IMA Journal of Applied Mathematics*, **24**(1):59–70, 08 1979. ()

[61] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. **Deep learning–based text classification: a comprehensive review**. *ACM computing surveys (CSUR)*, **54**(3):1–40, 2021. ()

[62] Zahra Mirzamomen and Mohammad Reza Kangavari. **Evolving Fuzzy Min–Max Neural Network Based Decision Trees for Data Stream Classification**. *Neural Processing Letters*, **45**(1):341–363, Feb 2017. ()

[63] M. F. Mohammed and C. P. Lim. **An Enhanced Fuzzy Min–Max Neural Network for Pattern Classification**. *IEEE Transactions on Neural Networks and Learning Systems*, **26**(3):417–429, 2015. ()

[64] Mohammed Falah Mohammed and Chee Peng Lim. **Improving the Fuzzy Min-Max neural network with a K-nearest hyperbox expansion rule for pattern classification**. *Applied Soft Computing*, **52**:135 – 145, 2017. ()

[65] Mohammed Falah Mohammed and Chee Peng Lim. **A new hyperbox selection rule and a pruning strategy for the enhanced fuzzy min–max neural network**. *Neural Networks*, **86**:69 – 79, 2017. ()

[66] A. V. Nandedkar and P. K. Biswas. **A Fuzzy Min-Max Neural Network Classifier With Compensatory Neuron Architecture**. *IEEE Transactions on Neural Networks*, **18**(1):42–54, 2007. ()

[67] A. V. Nandedkar and P. K. Biswas. **A Granular Reflex Fuzzy Min–Max Neural Network for Classification**. *IEEE Transactions on Neural Networks*, **20**(7):1117–1134, 2009. ()

[68] H. Son NGUYEN. **Discretization Methods in Data Mining**. *Rough Sets in Knowledge Discovery*, pages 451–482, 1998. ()

[69] Hung Son Nguyen. **Discretization Problem for Rough Sets Methods**. In Lech Polkowski and Andrzej Skowron, editors, *Rough Sets and Current Trends in Computing*, pages 545–552, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. ()

[70] Hung Son Nguyen. **Discretization Problem for Rough Sets Methods**. In Lech Polkowski and Andrzej Skowron, editors, *Rough Sets and Current Trends in Computing*, pages 545–552, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. ()

[71] Son H. Nguyen and Andrzej Skowron. **Quantization Of Real Value Attributes - Rough Set and Boolean Reasoning Approach**. In *Proc. of the Second Joint Annual Conference on Information Sciences, Wrightsville Beach, North Carolina, Sept 28 - Oct 1*, pages 34–37, 1995. ()

[72] Peng Ni, Suyun Zhao, Xizhao Wang, Hong Chen, and Cuiping Li. **PARA: A positive-region based attribute reduction accelerator**. *Information Sciences*, **503**:533–550, 2019. ()

[73] Peng Ni, Suyun Zhao, Xizhao Wang, Hong Chen, Cuiping Li, and Eric C.C. Tsang. **Incremental feature selection based on fuzzy rough sets**. *Information Sciences*, **536**:185 – 204, 2020. ()

[74] Sankar K. Pal, Biswarup Dasgupta, and Pabitra Mitra. **Rough Self Organizing Map**. *Applied Intelligence*, **21**(3):289–299, Nov 2004. ()

[75] N. M. Parthaláin, R. Jensen, and R. Diao. **Fuzzy-Rough Set Bireducts for Data Reduction**. *IEEE Transactions on Fuzzy Systems*, **28**(8):1840–1850, 2020. ()

[76] Z. Pawlak. **Rough sets and decision tables**. In Andrzej Skowron, editor, *Computation Theory*, pages 187–196, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg. ()

[77] Zdzisław Pawlak. **Rough sets**. *International Journal of Computer & Information Sciences*, **11**(5):341–356, Oct 1982. ()

[78] Zdzislaw Pawlak. *Rough sets - theoretical aspects of reasoning about data.*, **9**. Kluwer, 1991. ()

[79] Zdzisław Pawlak. **Some Issues on Rough Sets**. In James F. Peters, Andrzej Skowron, Jerzy W. Grzymała-Busse, Bożena Kostek, Roman W. Świniarski, and Marcin S. Szczuka, editors, *Transactions on Rough Sets I*, pages 1–58, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ()

[80] Zdzisław Pawlak and Andrzej Skowron. **Rough sets and Boolean reasoning**. *Information Sciences*, **177**(1):41 – 73, 2007. ()

[81] Farhad Pourpanah, Chee Peng Lim, Xizhao Wang, Choo Jun Tan, Manjeevan Seera, and Yuhui Shi. **A hybrid model of fuzzy min–max and brain storm optimization for feature selection and data classification**. *Neurocomputing*, **333**:440 – 451, 2019. ()

[82] Wenbin Qian, Wenhao Shu, and Changsheng Zhang. **Feature Selection from the Perspective of Knowledge Granulation in Dynamic Set-valued Information System**. *Journal of Information Science and Engineering*, **32**:783–798, 05 2016. ()

[83] Yuhua Qian, Qi Wang, Honghong Cheng, Jiye Liang, and Chuangyin Dang. **Fuzzy-rough feature selection accelerator**. *Fuzzy Sets and Systems*, **258**:61–78, 2015. Special issue: Uncertainty in Learning from Big Data. ()

[84] A. Quteishat, C. P. Lim, and K. S. Tan. **A Modified Fuzzy Min–Max Neural Network With a Genetic-Algorithm-Based Rule Extractor for Pattern Classification**. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, **40**(3):641–650, 2010. ()

[85] Anas Quteishat and Chee Peng Lim. **Application of the Fuzzy Min-Max Neural Networks to Medical Diagnosis**. In Ignac Lovrek, Robert J. Howlett, and Lakhmi C. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems*, pages 548–555, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ()

[86] Anas Quteishat and Chee Peng Lim. **A modified fuzzy min–max neural network with rule extraction and its application to fault detection and classification**. *Applied Soft Computing*, **8**(2):985 – 995, 2008. ()

[87] Anna Maria Radzikowska and Etienne E. Kerre. **A comparative study of fuzzy rough sets**. *Fuzzy Sets and Systems*, **126**(2):137 – 155, 2002. ()

[88] Orion F. Reyes-Galaviz and Witold Pedrycz. **Granular fuzzy modeling with evolving hyperboxes in multi-dimensional space of numerical data**. *Neurocomputing*, **168**:240 – 253, 2015. ()

[89] A. Rizzi, M. Panella, and F. M. Frattale Mascioli. **Adaptive resolution min-max classifiers**. *IEEE Transactions on Neural Networks*, **13**(2):402–414, 2002. ()

[90] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. **Learning representations by back-propagating errors**. *Nature*, **323**(6088):533–536, Oct 1986. ()

[91] P. Sadeghian, C. Wilson, S. Goeddel, and A. Olmsted. **Classification of music by composer using fuzzy min-max neural networks**. In *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 189–192, Dec 2017. ()

[92] P. S. V. S. Sai Prasad and C. Raghavendra Rao. *An Efficient Approach for Fuzzy Decision Reduct Computation*, pages 82–108. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ()

[93] Manjeevan Seera, M.L. Dennis Wong, and Asoke K. Nandi. **Classification of ball bearing faults using a hybrid intelligent model**. *Applied Soft Computing*, **57**:427 – 435, 2017. ()

[94] Manjeevan Seera, M.L. Dennis Wong, and Asoke K. Nandi. **Classification of ball bearing faults using a hybrid intelligent model**. *Applied Soft Computing*, **57**:427 – 435, 2017. ()

[95] Qiang Shen and Richard Jensen. **Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring**. *Pattern Recognition*, **37**(7):1351–1363, 2004. ()

[96] Wenhao Shu, Wenbin Qian, and Yonghong Xie. **Incremental feature selection for dynamic hybrid data using neighborhood rough set**. *Knowledge-Based Systems*, **194**:105516, 2020. ()

[97] Wenhao Shu and Hong Shen. **Incremental feature selection based on rough set in dynamic incomplete data**. *Pattern Recognition*, **47**(12):3890–3906, 2014. ()

[98] Wenhao Shu and Hong Shen. **Updating attribute reduction in incomplete decision systems with the variation of attribute set**. *International Journal of Approximate Reasoning*, **55**(3):867 – 884, 2014. ()

[99] F. Shuang and C. L. P. Chen. **Fuzzy restricted Boltzmann machine and deep belief network: A comparison on image reconstruction**. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1828–1833, Oct 2017. ()

[100] P. K. Simpson. **Fuzzy min-max classification with neural networks**. In *[1991 Proceedings] IEEE Conference on Neural Networks for Ocean Engineering*, **4**, pages 291–300, 1991. ()

[101] P. K. Simpson. **Fuzzy min-max neural networks**. In *[Proceedings] 1991 IEEE International Joint Conference on Neural Networks*, pages 1658–1669 vol.2, 1991. ()

[102] P. K. Simpson. **Fuzzy min-max neural networks. I. Classification**. *IEEE Transactions on Neural Networks*, **3**(5):776–786, 1992. ()

[103] Andrzej Skowron and Cecylia Rauszer. *The Discernibility Matrices and Functions in Information Systems*, pages 331–362. Springer Netherlands, Dordrecht, 1992. ()

[104] Dominik Slezak. **Approximate Reducts in Decision Tables**. In *Proc. Sixth Int'l Conf. Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 1159–1164, 1996. ()

[105] Roman Słowiński and Jerzy Stefanowski. **Handling Various Types of Uncertainty in the Rough Set Approach**. In Wojciech P. Ziarko, editor, *Rough Sets, Fuzzy Sets and Knowledge Discovery*, pages 366–376, London, 1994. Springer London. ()

[106] Preetee M. Sonule and Balaji S. Shetty. **An enhanced fuzzy min–max neural network with ant colony optimization based-rule-extractor for decision making**. *Neurocomputing*, **239**:204 – 213, 2017. ()

[107] Xiaoyan Sun, Lian Liu, Cong Geng, and Shaofeng Yang. **Fast Data Reduction With Granulation-Based Instances Importance Labeling**. *IEEE Access*, **7**:33587–33597, 2019. ()

[108] Roman W. Swiniarski and Andrzej Skowron. **Rough set methods in feature selection and recognition**. *Pattern Recognition Letters*, **24**(6):833–849, 2003. ()

[109] Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar. *Introduction to Data Mining (2nd Edition)*. Pearson, 2nd edition, 2018. ()

[110] K. Thangavel and A. Pethalakshmi. **Dimensionality reduction based on rough set theory: A review**. *Applied Soft Computing*, **9**(1):1–12, 2009. ()

[111] Chun-Wei Tsai, Chin-Feng Lai, Han-Chieh Chao, and Athanasios V. Vasilakos. **Big data analytics: a survey**. *Journal of Big Data*, **2**(1):21, Oct 2015. ()

[112] E. C. C. Tsang, D. Chen, D. S. Yeung, X. Wang, and J. W. T. Lee. **Attributes Reduction Using Fuzzy Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **16**(5):1130–1141, 2008. ()

[113] C. Wang, Y. Qi, M. Shao, Q. Hu, D. Chen, Y. Qian, and Y. Lin. **A Fitting Model for Feature Selection With Fuzzy Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **25**(4):741–753, 2017. ()

[114] Feng Wang, Jiye Liang, and Chuangyin Dang. **Attribute reduction for dynamic data sets**. *Applied Soft Computing*, **13**(1):676 – 689, 2013. ()

[115] Feng Wang, Jiye Liang, and Yuhua Qian. **Attribute reduction: A dimension incremental strategy**. *Knowledge-Based Systems*, **39**:95 – 108, 2013. ()

[116] X. Wang, F. He, and L. Liu. **Application of Rough Set Theory to Intrusion Detection System**. In *2007 IEEE International Conference on Granular Computing (GRC 2007)*, pages 562–562, 2007. ()

[117] Wei Wei, Xiaoying Wu, Jiye Liang, Junbiao Cui, and Yijun Sun. **Discernibility matrix based incremental attribute reduction for dynamic data**. *Knowledge-Based Systems*, **140**:142 – 157, 2018. ()

[118] Y. Yang, D. Chen, and H. Wang. **Active Sample Selection Based Incremental Algorithm for Attribute Reduction With Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **25**(4):825–838, Aug 2017. ()

[119] Y. Yang, D. Chen, H. Wang, and X. Wang. **Incremental Perspective for Feature Selection Based on Fuzzy Rough Sets**. *IEEE Transactions on Fuzzy Systems*, **26**(3):1257–1273, June 2018. ()

[120] Yanyan Yang, Degang Chen, Hui Wang, Eric C.C. Tsang, and Deli Zhang. **Fuzzy rough set based incremental attribute reduction from dynamic data with sample arriving**. *Fuzzy Sets and Systems*, **312**:66 – 86, 2017. Theme: Fuzzy Rough Sets. ()

[121] Jing Tao Yao, Athanasios V. Vasilakos, and Witold Pedrycz. **Granular Computing: Perspectives and Challenges**. *IEEE Transactions on Cybernetics*, **43**(6):1977–1989, 2013. ()

[122] Yiyu Yao and Yan Zhao. **Discernibility matrix simplification for constructing attribute reducts**. *Information Sciences*, **179**(7):867 – 882, 2009. ()

[123] Yiyu Yao, Yan Zhao, and Jue Wang. **On Reduct Construction Algorithms**. In Guo-Ying Wang, James F. Peters, Andrzej Skowron, and Yiyu Yao, editors, *Rough Sets and Knowledge Technology*, pages 297–304, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ()

[124] D. S. Yeung, Degang Chen, E. C. C. Tsang, J. W. T. Lee, and Wang Xizhao. **On the generalization of fuzzy rough sets**. *IEEE Transactions on Fuzzy Systems*, **13**(3):343–361, 2005. ()

[125] L. A. Zadeh. *FUZZY SETS AND INFORMATION GRANULARITY*, pages 433–448. ()

[126] L.A. Zadeh. **Fuzzy sets**. *Information and Control*, **8**(3):338 – 353, 1965. ()

[127] L.A. Zadeh. **Fuzzy Sets and Information Granularity**. Technical Report UCB/ERL M79/45, EECS Department, University of California, Berkeley, Jun 1979. ()

[128] L.A. Zadeh. **Key Roles of Information Granulation and Fuzzy Logic in Human Reasoning, Concept Formulation and Computing with Words**. In *Proceedings of IEEE 5th International Fuzzy Systems*, **1**, pages 1–1, 1996. ()

[129] L.A. Zadeh. **Key Roles of Information Granulation and Fuzzy Logic in Human Reasoning, Concept Formulation and Computing with Words**. In *Proceedings of IEEE 5th International Fuzzy Systems*, **1**, pages 1–1, 1996. ()

[130] Anping Zeng, Tianrui Li, Dun Liu, Junbo Zhang, and Hongmei Chen. **A fuzzy rough set approach for incremental feature selection on hybrid information systems**. *Fuzzy Sets and Systems*, **258**:39 – 60, 2015. Special issue: Uncertainty in Learning from Big Data. ()

[131] H. Zhang, J. Liu, D. Ma, and Z. Wang. **Data-Core-Based Fuzzy Min–Max Neural Network for Pattern Classification**. *IEEE Transactions on Neural Networks*, **22**(12):2339–2352, 2011. ()

[132] Junbo Zhang, Tianrui Li, and Hongmei Chen. **Composite rough sets for dynamic data mining**. *Information Sciences*, **257**:81–100, 2014. ()

[133] Junbo Zhang, Tianrui Li, Da Ruan, and Dun Liu. **Rough sets based matrix approaches with dynamic attribute variation in set-valued information systems**. *International Journal of Approximate Reasoning*, **53**(4):620 – 635, 2012. ()

[134] Shichao Zhang, Chengqi Zhang, and Qiang Yang. **Data preparation for data mining**. *Applied Artificial Intelligence*, **17**(5-6):375–381, 2003. ()

[135] X. Zhang, C. Mei, D. Chen, Y. Yang, and J. Li. **Active Incremental Feature Selection Using a Fuzzy-Rough-Set-Based Information Entropy**. *IEEE Transactions on Fuzzy Systems*, **28**(5):901–915, 2020. ()

[136] Xiao Zhang, Xia Liu, and Yanyan Yang. **A Fast Feature Selection Algorithm by Accelerating Computation of Fuzzy Rough Set-Based Information Entropy**. *Entropy*, **20**(10), 2018. ()

[137] Xiao Zhang, Changlin Mei, Degang Chen, and Jinhai Li. **Feature selection in mixed data: A method using a novel fuzzy rough set-based information entropy**. *Pattern Recognition*, **56**:1–15, 2016. ()

[138] Xiao Zhang, Changlin Mei, Degang Chen, and Yanyan Yang. **A fuzzy rough set-based feature selection method using representative instances**. *Knowledge-Based Systems*, **151**:216 – 229, 2018. ()

[139] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. **Fuzzy deep belief networks for semi-supervised sentiment classification**. *Neurocomputing*, **131**:312 – 322, 2014. ()

[140] Wojciech Ziarko. **Rough Sets and Knowledge Discovery: An Overview**. In Wojciech P. Ziarko, editor, *Rough Sets, Fuzzy Sets and Knowledge Discovery*, pages 11–15, London, 1994. Springer London. ()

[141] H.J. Zimmermann. *Fuzzy Sets and Expert Systems*, pages 185–222. Springer Netherlands, Dordrecht, 2001. ()

[142] S. Zobeidi, M. Naderan, and S. E. Alavi. **Effective text classification using multi-level fuzzy neural network**. In *2017 5th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*, pages 91–96, March 2017. ()

[143] QIN ZOU, LIHAO NI, TONG ZHANG, AND QIAN WANG. **Deep learning based feature selection for remote sensing scene classification**. *IEEE Geoscience and Remote Sensing Letters*, **12**(11):2321–2325, 2015. ()

[144] ALEKSANDER ØHRN. **Discernibility and Rough Sets in Medicine: Tools and Applications**. *Norwegian University of Science and Technology*, 01 2000. ()